НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра технічної кібернетики

«На правах рукопису»
УДК *004.043*

«До захисту допущено»

Завідувач кафедри
_____ І.Р. Пархомей
(підпис)
“___”_____2019 р.

Магістерська дисертація

на здобуття ступеня магістра

зі спеціальності 126 «Інформаційні системи та технології»

на тему: Рейтингова система навчання студентів з дисципліни робототехніка

Виконав: студент другого курсу, групи ІК-81мп
(шифр групи)

Вєтошкін Ігор Володимирович _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник ст. викладач Анікін В.К. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант НК доцент Пасько В.П. _____
(назва розділу) (науковий ступінь, вчене звання, , прізвище, ініціали) (підпис)

Рецензент _____ _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2019 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра технічної кібернетики

Рівень вищої освіти – другий (магістерський)

Спеціальність 126 «Інформаційні системи та технології»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ І.Р. Пархомей
(підпис)
«___»_____2019 р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Вєтошкіну Ігорю Володимировичу
(прізвище, ім'я, по батькові)

1. Тема дисертації «Рейтингова система навчання студентів з дисципліни робототехніка»,
науковий керівник дисертації ст. викладач Анікін В.К.,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «28» жовтня 2019 р. № 3770-с

2. Термін подання студентом дисертації 18.11.2019

3. Об'єкт дослідження – *процес роботи рейтингової системи у веб-додатку для навчання студентів.*

4. Предмет дослідження – *методи та технології оптимізації роботи веб-додатку, які б забезпечили його належну швидкодію.*

5. Перелік завдань, які потрібно розробити – *аналіз існуючих рішень; вибір технологій для розробки системи та їх опис; розробка системи для рейтингу студентів; оптимізація роботи системи та забезпечення її належної швидкодії; тестування системи.*

6. Орієнтовний перелік ілюстративного матеріалу – 6 плакатів

7. Орієнтовний перелік публікацій – 2 публікації

8. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
НК	Пасько В.П. доцент		
Перевірка на співпадіння	Лісовиченко О.І. доцент		

9. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Аналіз предметної області	03.09.2019 – 07.09.2019	
2	Постановка задачі	8.09.2019 – 11.09.2019	
3	Вибір технологій для розробки системи	12.09.2019 – 22.09.2019	
4	Проектування бази даних	23.09.2019 – 29.09.2019	
5	Розробка основних модулів	30.09.2019 – 10.10.2019	
6	Оптимізація роботи системи	11.10.2019 – 19.10.2019	
7	Тестування розробленої системи	20.10.2019 – 23.10.2019	
8	Практичне застосування	24.10.2019 – 27.10.2019	

Студент _____ Вєтошкін І.В.
(підпис) (ініціали, прізвище)

Науковий керівник дисертації _____ Анікін В.К.
(підпис) (ініціали, прізвище)

АНОТАЦІЯ

В цій роботі розглянуто проблему у області оптимізації веб-додатків, а саме оптимізація рейтингової системи.

Проаналізовано аналоги рейтингових систем для навчання студентів та інформаційні системи, які ознайомлюють користувачів з навчальними закладами, визначено їх основні переваги та недоліки. На основі отриманих результатів визначені основні необхідні модулі, які повинна містити дана система, способи її оптимізації та забезпечення належної швидкодії.

В результаті виконання дипломної дисертації розроблено рейтингову систему для навчання студентів. В рамках веб-додатку, здійснено її оптимізацію та значно підвищено швидкодію за рахунок оптимізації структури бази даних, використання фреймворку React в розробці клієнтської частини для створення SPA, впровадження додаткової системи кешування даних, оптимізації запитів до бази даних.

Ключові слова: оптимізація швидкодії роботи рейтингової системи, рейтингова система для навчання, кешування даних, SPA, React, MVC.

Розмір пояснювальної записки – 88 сторінок, 48 ілюстрацій, 22 таблиці та 2 додатки.

ABSTRACT

This master thesis shows the problem of web application optimization, describes the optimization of the rating system.

Analogues of rating systems for student education and information systems were analyzed, their main advantages and disadvantages were identified. On the basis of the obtained results, the basic necessary modules, which should be contained by the given system, ways of its optimization and ensuring proper performance, were created.

As a result of the master thesis completion, a rating system for learning students was developed. In the web application, optimization was completed and performance was significantly increased due to optimization of the database structure, using of React framework in the development of the client side for creating SPA, implementation of additional data caching system, optimization database queries.

Keywords: performance optimization of rating system, rating system for teaching, data caching, SPA, React, MVC.

Explanatory note size – 88 pages, contain 48 illustrations, 22 tables and 2 applications.

ПОЯСНЮВАЛЬНА ЗАПИСКА
до магістерської дисертації

на тему: *Рейтингова система навчання студентів з дисципліни робототехніка*

Зміст

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

HTTP – hypertext transfer protocol

JS – JavaScript

ES – ECMAScript

DB – Database

CSS – Cascade Style Sheet

API – application programming interface

JSON – JavaScript object notation

СУБД – система управління базою даних

HTML – Hypertext Markup Language

ВСТУП

В наш час велике значення в житті кожної людини має інтернет. Ця мережа має настільки великі масштаби, що може надати інформацію за будь-яким запитом за лічені секунди. Велика кількість послуг та задач представлені у цифровому форматі та виконуються за допомогою певного веб-додатку. За останні роки з'явилася велика кількість навчальних закладів та студій, які мають електронну програму навчання з різних сфер знань. Переваги таких програм в вивченні студентом виключно тієї інформації, яка є актуальною та необхідною для студента. Ще один фактор, який присутній в таких програмах – це відбір предметів, які студенту важливо та цікаво вчити. На жаль більшість державних навчальних закладів мають застарілу базу знань, яку студент навряд чи зможе застосувати на практиці в майбутньому. Програма в цифрових навчальних закладах може максимально кастомізуватися, що дозволяє студенту вивчати тільки те, що йому подобається.

Веб-студії та компанії, які надають репетиторські послуги частіше всього мають зворотню ситуацію. Вони володіють актуальним набором знань, сучасними практиками, але не завжди вони мають певну систему, яка автоматизує цей процес.

З розвитком цього напрямку навчання розвивалися й компанії, які надають такі послуги. На сьогоднішній день ми маємо велику кількість веб-студій та навчальних закладів, створено багато платформ, які надають можливість вивчення різних областей знань. Більшість з них має свої веб-додатки з електронним кабінетом, в якому є велика кількість інформації, яка необхідна студенту. Особливістю таких веб-студій часто є гарантія майбутнього трудовлаштування. Однією з моделей співпраці навчального закладу та роботодавця є наступне: створюється рейтингова система, яка орієнтується на базу завдань різної складності. Проходячи завдання студенти отримують певні бали. таким чином формується рейтингова система. Компанії роботодавців можуть мати свої критерії відбору та свої тестові завдання.

Завдання матимуть певну градацію по рівню знань, для максимально якісного висвітлення рівня знань, того хто виконує. Виконуючи завдання, студенти збільшують свою рейтингову оцінку та просуваються вгору за рейтинговим списком. До завдань, на яких буде ґрунтуватись рейтинг студента матиме доступ потенційний роботодавець. Додаючи своє завдання до бази завдань, він може відслідковувати студентів, які виконали завдання та запрошувати на роботу до своєї компанії. Таким чином створюється єдина база студентів, які можуть визначити свій рівень знань та знати актуальні вимоги ІТ компаній.

Мета даної магістерської роботи – оптимізація та підвищення швидкодії роботи рейтингової системи навчання студентів з дисципліни робототехніка; система, яка матиме можливість оцінки рівня знань студентів за виконані завдання різної складності.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ І ПОСТАНОВКА ЗАДАЧІ

1.1. Об’єкт та предмет дослідження

Для даної магістерської дисертації було проведено аналіз інформації в області створення та оптимізації веб-додатків. На сьогоднішній день ми маємо велику кількість веб-додатків пов’язаних з наданням навчальних послуг в тому або іншому вигляді. Загалом, можна виділити декілька видів недоліків в таких системах. Це перевантаженість даними, що зменшує сприйняття інтерфейсу користувача та ускладнює розуміння матеріалів, які розміщені на таких ресурсах. Головне правило, до якого необхідно прислухатися – це контроль вільного простору. Маючи весь екран для розміщення інформації, не треба намагатися повністю його заповнити. За допомогою вільного простору навколо блоку інформації можна структурувати контент та акцентувати увагу на певних блоках інформації. Наступний недолік – це швидкодія додатку. Через великий об’єм даних система завантажується довше та змушує користувача чекати. Дослідження продемонстрували, що п’ятдесят три відсотки людей закриють сторінку браузера, яка вона буде завантажуватись довше трьох секунд[1]. З описаних вище проблем можна зробити висновок, що існує потреба в розробці рейтингової системи навчання студентів, яка має зручний та ергономічний інтерфейс, належну швидкодію та певну інтерактивність, яка у випадку низької швидкодії(погане з’єднання в мережі інтернет, інші види збоїв) демонструватиме стан сторінки, яка завантажується. Рейтингова система повинна коректно відображати результати роботи студентів над певними завданнями. Кількість студентів, які будуть оцінюватись в одній рейтинговій групі на створюється та редагується розсуд адміністратора ресурсу. Кількість даних, які існуватимуть в рамках цієї системи теж залежить від навчальної програми компанії, яка буде користуватись системою. Тому треба створити інтерфейс, який створить зручне представлення усієї інформації в системі. Також, необхідно придати

увагу ситуації коли багато користувачів одночасно зайшли на сторінку і таким чином створили велику кількість запитів до серверу. Для досягнення високих показників швидкодії та низької кількості помилок та багів система повинна бути протестована.

Об'єкт дослідження – процес роботи веб-додатку рейтингової системи навчання студентів з дисципліни робототехніка, швидкодія роботи системи, швидкість відпрацювання запитів користувачів.

Предмет дослідження – методи та технології оптимізації роботи веб-додатку, які б забезпечили його належну швидкодію.

1.2. Аналіз існуючих систем

Приступаючи до аналізу треба встановити які системи будуть прийматися до уваги і що саме треба оцінити в цих системах. Головною тематикою систем, які будуть розглядатися буде навчання студентів або курси навчання певним сферам навчання. Фактори, які варто оцінити – це дизайн інтерфейсу та його зручність, наявність аналогів рейтингової системи та наявність в них схожого функціоналу та, звісно, швидкодія роботи системи.

Дизайн інтерфейсу – один з найважливіших факторів системи. Він відповідає не тільки за роботу системи, а й за його маркетингову складову. Відомий факт, що кожна людина оцінює спочатку візуальну складову. Аналогічна ситуація і з дизайном, якщо існують системи, які виконують коректно поставлену функціональну задачу, то скоріш за все користувачі будуть давати перевагу системі з більш сучасним та красивим дизайном.

Наступний фактор, який впливає на роботу користувача з системою це зручність системи. Система не повинна бути завантаженою інформацією. Користувач не зможе швидко орієнтуватись в необхідній йому інформації та буде затримуватись на кроках, на які він взагалі не повинен звертати увагу. Прикладом гарного інтерфейсу можна вважати інтерфейс, виконаний за «правилом трьох». Коли користувач може дістатися до будь-якої інформації

на сайті в три кліки миші. Частіше за все це реалізується за рахунок правильно створеної структури навігації по сайту.

Швидкодія роботи системи – головна складова нашої роботи. Тому необхідно дослідити за допомогою яких технологій був досягнений той чи інший результат та як можна його покращити.

При розгляді аналогів, які вже існують на ринку необхідно оглянути весь функціонал, які вони мають. Звісно, якщо дані веб-додатки мають вже розроблені рейтингові системи, то необхідно їх проаналізувати, зробити певні висновки та продумати особливості, за допомогою яких можна конкурувати з існуючими аналогами.

Розглянемо одну з найпопулярніших систем в нашій країні ITVDN (рис. 1.1).

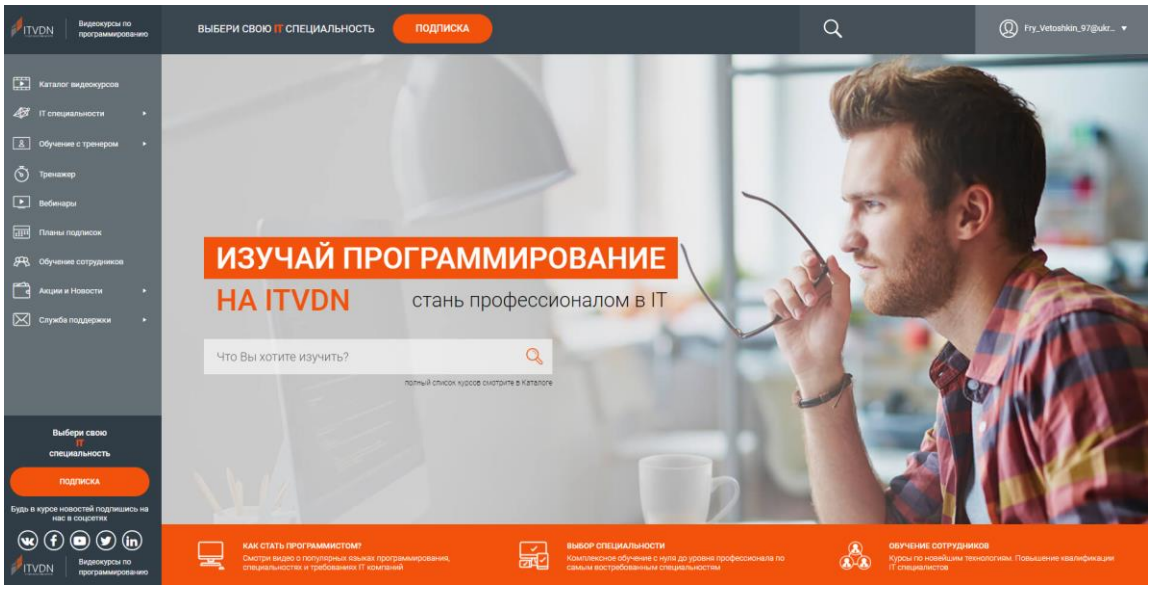


Рисунок 1.1. Система для навчання ITVDN

Це велика веб-студія, яка надає навчальні послуги у вигляді курсів, займається як онлайн, так і офлайн курсами. Дана система має сучасний дизайн у мінімалістичному та модульному стилі. Також прослідкується F-модель у дизайні (рис. 1.3) [2]. Це варіант створення сайту, враховуючи особливості сприйняття людиною інформації. Було проведено дослідження та за допомогою термографічної експертизи визнали, що людина читаючи веб

сторінку, уявно розділяє її на лінії. Таким чином, людина легше сприймає інформацію, яка подана у вигляді F-моделі (рис. 1.2). Також важливу частину в дизайні інтерфейсів займають так звані якірні точки, які теж привертають увагу користувача. Такими точками є кути сторінки, та кути різних прямокутних елементів на сайті. Тому часто в такі точки вставляють необхідну для клієнта інформацію: навігація, логотип, ключові кнопки(передзвонити, залишити заявку і т.д.). Також, ця модель орієнтується на національні особливості, за якими люди читають текст. Наприклад, в арабських країнах пишуть справа наліво, тому F-модель сайту відображується дзеркально по горизонталі (логотип та навігація сайту в правому краю сторінки, і т.д).

Рисунок 1.2. Теплова картина, яка відображує куди людина на сайті в основному дивиться

На сайті ITVDN відслідковується використання F-модель. Можна побачити чітко, що в лівому верхньому куті розміщений логотип-посилання сайту, в правому верхньому куті розміщені кнопки пошуку та кнопку-посилання на особистий кабінет, по лівому боку розміщена навігація по сайту.

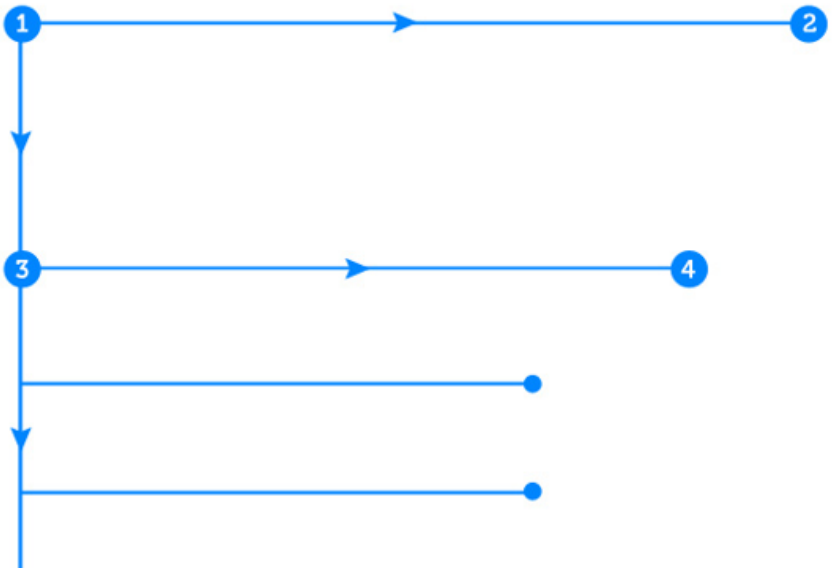


Рисунок 1.3. F-Модель структури сайту

Головна сторінка сайту виглядає достатньо лаконічно. Сайт має логотип, зафіксовану навігацію зліва та основний блок інформації. На головній сторінці в основному блоці, який людина бачить при відкритті сайту є тематичне зображення та поле пошуку за цікавими курсами. Далі в основному блоці є коротка інформація про послуги, які надає ця компанія та переваги, які отримує клієнт навчаючись в цій компанії. Сторінка з популярними курсами виглядає також мінімалістично. На сайті виведені 6 курсів з різних напрямів з короткою інформацією по курсу (назва курсу, вчитель, кількість уроків та їх про тривалість).

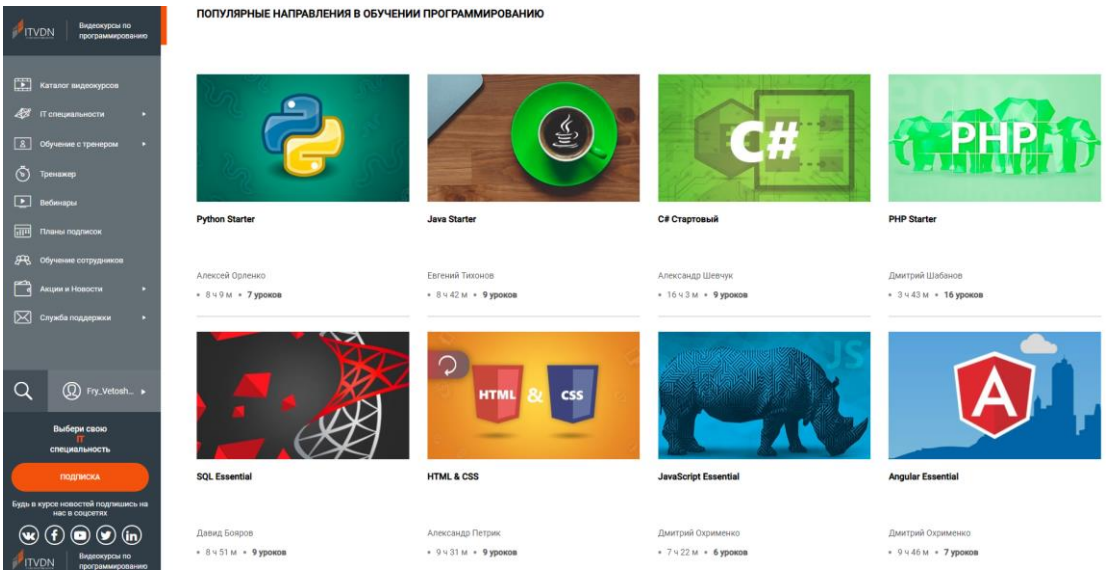


Рисунок 1.4. Перегляд популярних курсів на сайті

Після популярних курсів можна побачити блок з новинами , які пов’язані з успіхами компанії або просто новини зі сфери інформаційних технологій. За допомогою кнопки-перемикача можна перейти на розділ «Акції» де відповідно знаходяться архів старих акцій та акції, які діють на сьогоднішній день. Далі можна побачити блок відгуків від людей, які успішно завершили проходження курсів та мають певні успіхи на сьогоднішній день.

Далі був проаналізований особистий кабінет веб-сайту ITVDN. На мою думку, кабінет має складний дизайн, який забирає увесь вільний простір і нагромаджує необхідну інформацію. Можемо побачити блок з закладками, де розміщені усі відеоуроки, які цікавлять користувача. Кабінет має посилання на пройдені тести, перевірку домашніх завдань, акційні пропозиції та інформацію про особисті дані користувача.

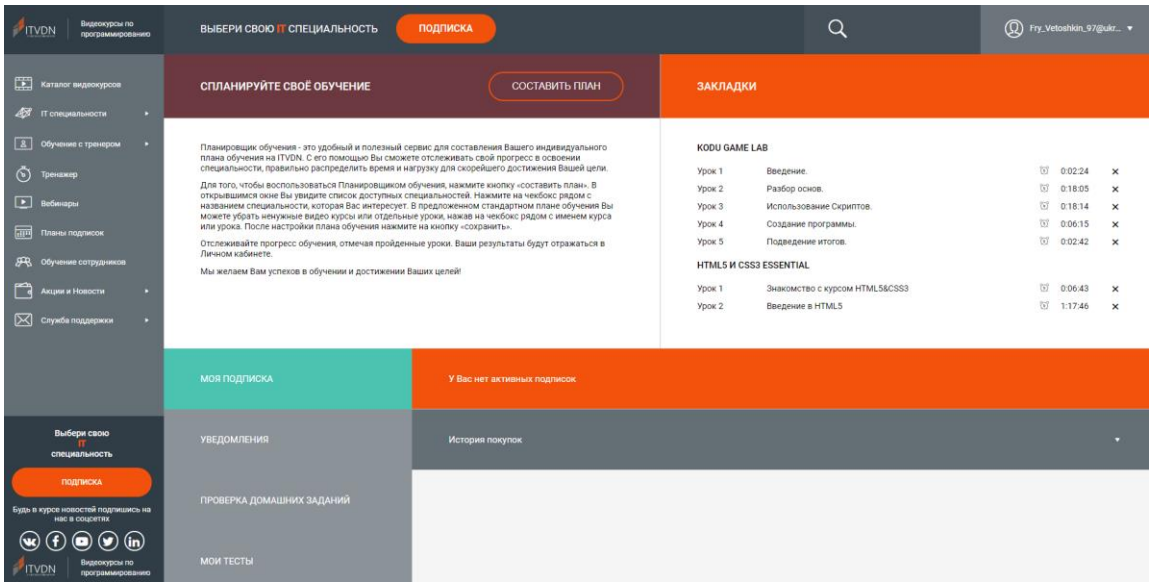


Рисунок 1.5. Особистій кабінет на сайті ITVDN

Нових користувачів зустрічає сторінка опитування про ресурси з яких користувач дізнався про веб-студію. На мою думку, сторінка виглядає занадто завантажено. Кращим варіантом було б зробити опитування за допомогою модального вікна, відкрити яке можна за допомогою кліку по звичайній кнопці. Таким чином звільниться місце під більш необхідну для користувача інформацію. Дизайн кнопок та блоків виконаний за допомогою великого

масштабу. Так як усі блоки мають великі розміри, заголовкам та указівкам не надається акцент, тому певна необхідна інформація втрачається в полі зору, клієнту потрібно більше часу для знаходження потрібної інформації. Посилання на соціальні мережі розташовані в зафіксованому блоці зліва, під усіма пунктами навігації. При розміщенні великої кількості інформації в основному блоці сайту, такий спосіб розміщення є дуже зручний.

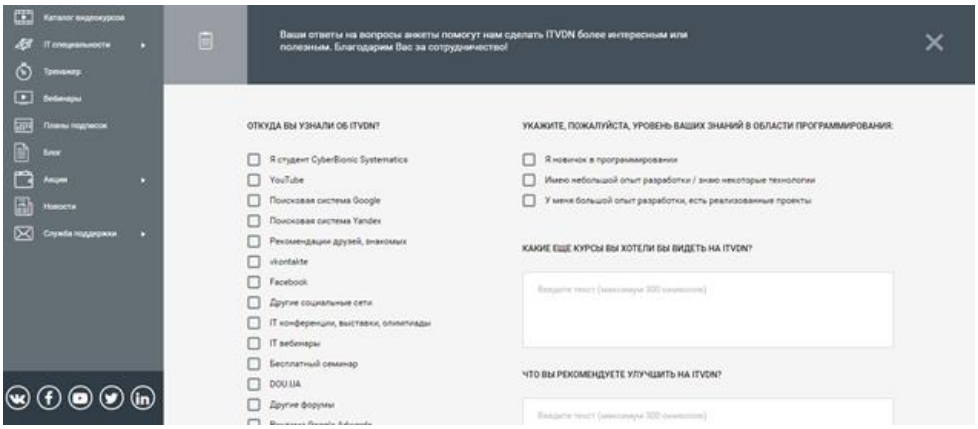


Рисунок 1.6. Особистій кабінет на сайті ITVDN

Розглянемо показники швидкодкодії сайту ITVDN (рис 1.7). За основу будемо вважати дані, які надає браузер Google Chrome через Google Chrome Developer Tools. Зайдемо в вкладку Network в панель розробника та проаналізуємо що завантажується та час за який завантажується сторінка. Серед всієї інформації можна побачити які файли завантажуються, їх статус при завантаженні, пріоритетність завантаження файлів та графік завантаження скриптів в часі. Також, будемо використовувати вкладку Console для перевірки на наявність небажаних коментарів, які залишилися після тестування функціоналу сайту або повідомлень про помилки.

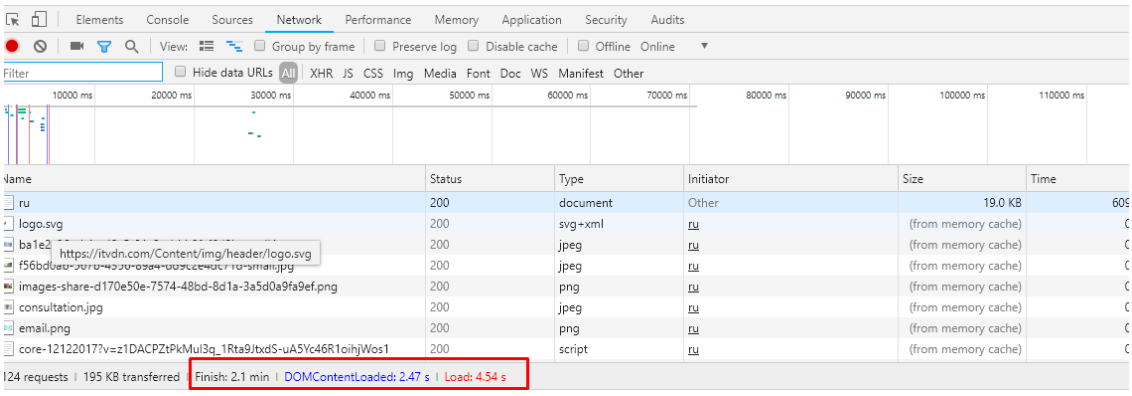


Рисунок 1.7. Час завантаження головної сторінки ITVDN

Перше завантаження головної сторінки сайту займає приблизно 5 секунд. На головній сторінці завантажуються певна текстова інформація, блоки із зображеннями, іконки та gif-анімація. На всьому сайті також завантажуються шрифти, скрипти для відслідковування SEO інформації та просування сайту в пошукових системах за допомогою Google Analytics. На сторінках з відеоматеріалами показники гірші, за рахунок завантаження великої кількості відеофайлів. Час за який на цьому сайті завантажується увесь контент складає 2 хвилини. Причиною такого довгого завантаження сторінки може бути велика кількість скриптів та велика кількість помилок в коді (можемо це побачити в консолі розробника – рис. 1.8). Можемо побачити, що в консолі відображені помилки не тільки про доступ до зовнішніх ресурсів, а й помилка в роботі певного функціоналу із внутрішніх скриптів.

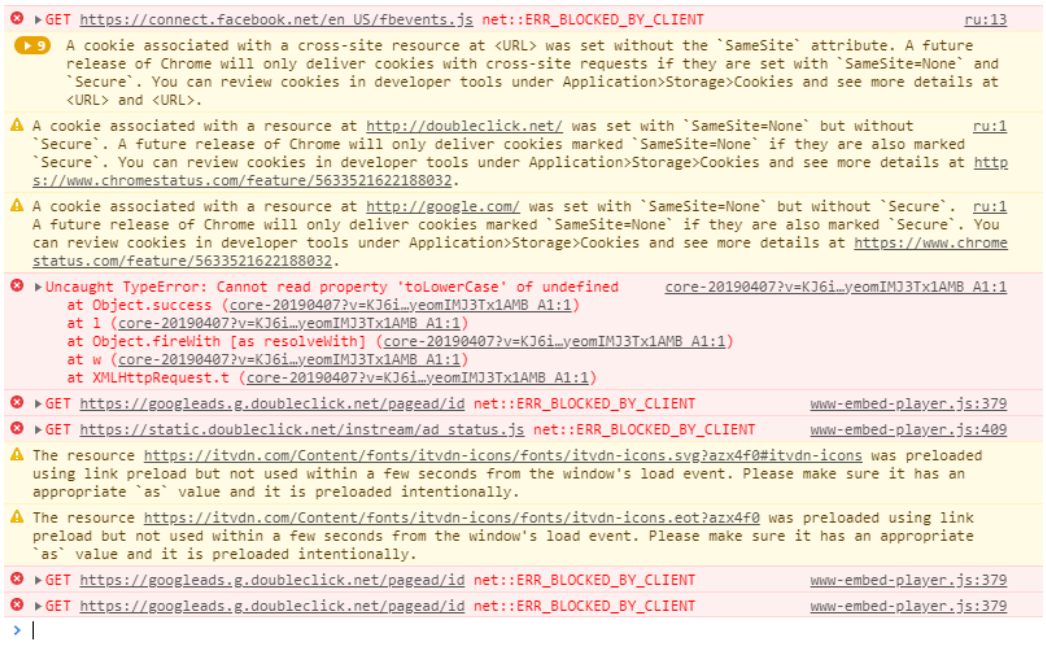


Рисунок 1.8. Консоль головної сторінки сайту ITVDN

В особистому кабінеті та на просторах сайту не було знайдено рейтингової системи та систем, які схожі за функціоналом. Таким чином, на даний момент можемо зробити висновок, що провідні системи не мають аналогів такого типу системи.

Наступною системою, яку ми будемо розглядати – це інформаційна система компанії Web Academy (рис. 1.9). Дана система також є одним з найкращих рішень для нашого регіону.

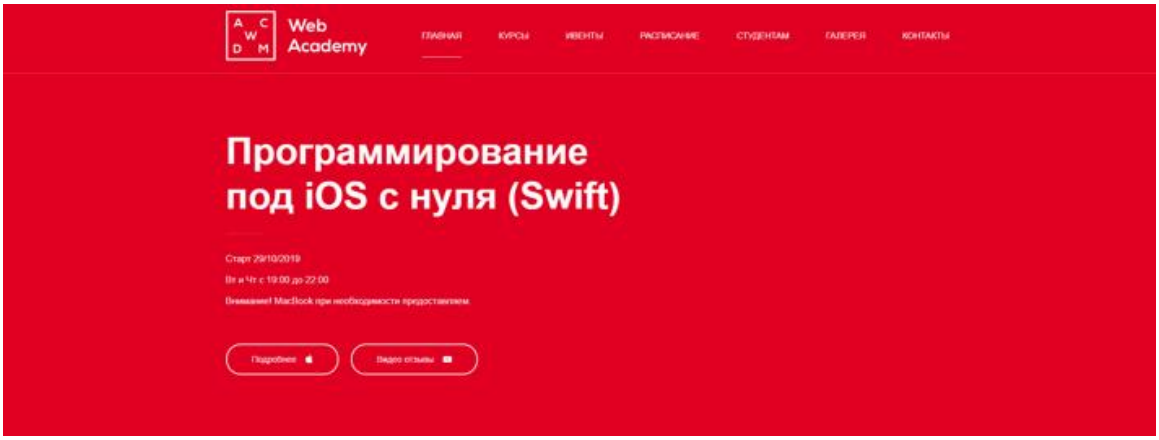


Рисунок 1.9. Головна сторінка системи для навчання Web Academy

На головній сторінці сайту Web Academy можна побачити мінімалістичний дизайн, просту палітру кольорів (червоний, білий та чорний для шрифтів). На першому екрані зображений слайдер з курсами, які починаються в найближчий час. Далі можна побачити список курсів та цін, та популярні направлення академії(рис 1.10). В таблиці курсів можна побачити назву напряму курсу, тип проходження курсу (онлайн або офлайн), дату початку курсу, розклад, ціну на курс та кнопка швидкого запису на курс.

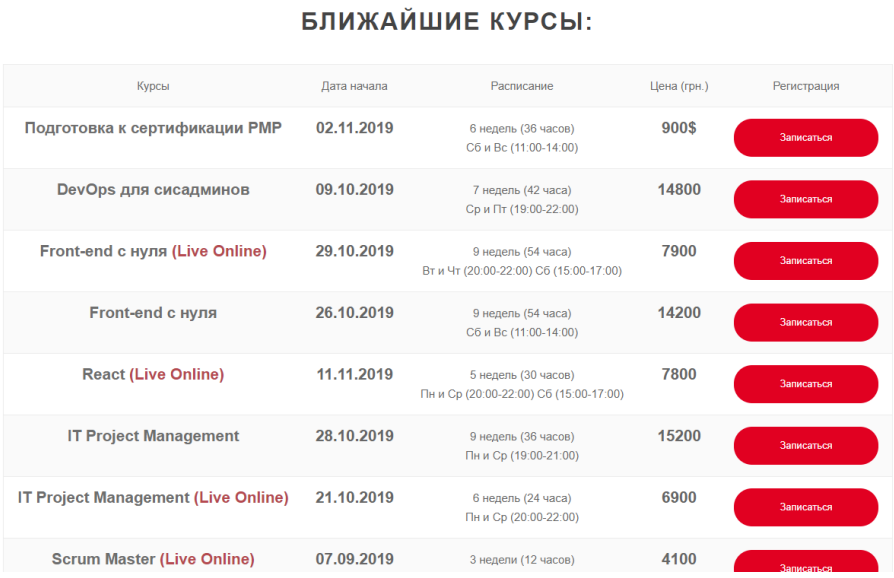


Рисунок 1.10. Сторінка системи Web Academy з таблицею курсів

Розглядаючи вкладку консоль в браузері можемо побачити велику кількість повідомлень про помилки. Також, наявні помилки, які пов’язані з маніпуляціями з DOM деревом.

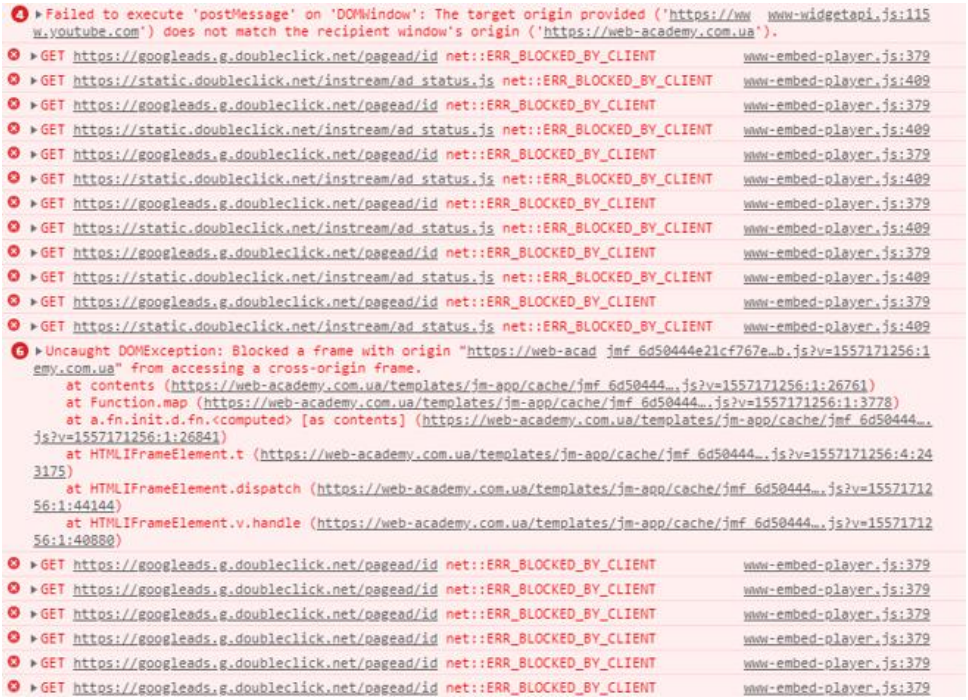


Рисунок 1.10. Помилки в консолі сторінки системи для навчання Web Academy

Даний сайт теж має проблеми з швидкодією сайту. Головна сторінка завантажується майже за 5 з половиною секунд. Увесь контент на сторінці завантажується за 2 хвилини. Також, в консолі розробника можна побачити велику кількість помилок. Значна частина помилок створена допоміжними скриптами, такими як Google Analytics та інші. Такі скрипти збільшують загальну швидкість завантаження сайту через періодичність відправлення запитів. Ще одним джерелом помилок є певний скрипт, доступ до якого частково заблокований. Основна задача цього скрипту – це забезпечення роботи програвача відеоматеріалів.

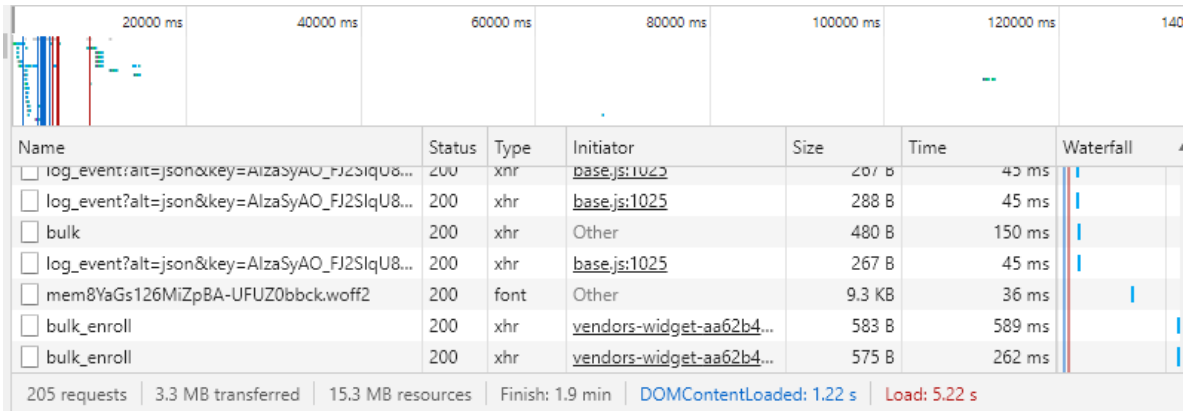


Рисунок 1.11. Час завантаження сторінки системи для навчання Web Academy

Ще одним недоліком даної системи є відсутність особистого кабінету та відсутність можливості переглядати певний курс онлайн за допомогою цього сайту. Уся інформація про результати проходження курсів відсутня в електронному вигляді. Відповідно, на сайті Web Academy відсутня рейтингова система. Система, яка представлена на цьому сайті має малу функціональність та не дає можливості отримувати інформацію по певному курсу в онлайн режимі та за допомогою відеоматеріалів. Рейтингова система на даному сайті також відсутня. Головна задача цього сайту – це можливість ознайомитись з компанією, яка надає курси, переглянути існуючі курси, ціну, але тип навчання, який надає дана система тільки у вигляді живого навчання з вчителем.

1.3. Постановка задачі

Метою даної магістерської дисертації є оптимізація та підвищення швидкодії рейтингової системи навчання студентів з дисципліни робототехніка. Для досягнення даної мети необхідно вирішити наступні задачі:

– зробити аналіз аналогів, які вже існують, виявити гарні та погані сторони в роботі з цим аналогом та визначити причини, які зменшують швидкість роботи додатку.

– визначити структуру додатку, основні модулі та компоненти системи на основі аналізу, який був проведений раніше;

– створити структуру бази даних із врахуванням найчастіше виконуваних запитів користувачів;

– написати REST API та логіку роботи основних модулів системи;

– впровадити систему кешування для підвищення швидкодії системи;

– покрити функціонал системи за допомогою unit-тестів;

– провести мануальне тестування системи;

– забезпечити масштабування системи;

Висновки до розділу

В цьому розділі для отримання необхідної інформації щодо систем дистанційного навчання було проаналізовано декілька існуючих систем навчання в мережі інтернет. Також була поставлена задача знайти аналоги рейтингової системи навчання. Однієї з систем, яка приймалась до уваги і яка була проаналізована є ITVDN. Система залишила після себе неоднозначне враження. Система має сучасний дизайн та використовує сучасні техніки створення зручних інтерфейсів користувача, але разом з тим дуже перенасичена інформацією. Особистий кабінет виглядає дуже загромадженим і незручним. Новий користувач одразу отримує опитувальник на всю сторінку з дуже великою запитань. Такий варіант отримання зворотнього зв’язку навряд буде дуже популярним і зможе надати чіткі дані за якими можна робити висновки для подальшої маркетингової роботи. Швидкодія системи теж не на високому рівні. Головна сторінка завантажується за 5 секунд, увесь контент завантажується за 2 хвилини. Також система має багато помилок, які виводяться в консоль. Вони є одним з причин такого довгого завантаження сторінки. Рейтингова система, яка нам необхідна для аналізу аналогу відсутня. Наступною системою, яка була розглянута стала Web Academy. Система має мінімалістичний дизайн та досить гармонічну палітру кольорів. На головній

сторінці зображена необхідна інформація, навігація виглядає зрозуміло. Проте, на сайті відсутній особистий кабінет користувача та немає рейтингової системи усіх користувачів, які навчаються. Швидкодія теж не на належному рівні (5 секунд на завантаження основної інформації та 2 хвилини на завантаження усього контенту). Враховуючи всі проаналізована існуючі системи було визначено мету, предмет та об’єкт дослідження. Визначено основні задачі, які потрібно виконати для досягнення даної мети.

РОЗДІЛ 2. ВИБІР ТЕХНОЛОГІЙ ДЛЯ РОЗРОБКИ РЕЙТИНГОВОЇ СИСТЕМИ НАВЧАННЯ СТУДЕНТІВ З ДИСЦИПЛІНИ РОБОТОТЕХНІКА

Маючи на меті створити рейтингову систему, яка буде використовуватись у вигляді веб-додатку ми повинні визначитись з технологіями розробки клієнтської та серверної частини. Для клієнтської частини для розмітки буде використовуватись HTML, CSS. Для створення певного функціоналу на веб-сторінках будемо використовувати мову програмування JavaScript. Для підвищення швидкодії додатку будемо використовувати фреймворк React та додаткові бібліотеки, яку фреймворк буде використовувати. Для створення стилів до веб-сторінок можна долучити фреймворк Bootstrap. На даний момент він є одним з найпопулярніших інструментів з готовими стильовими рішеннями. Зокрема, створення розмітки у вигляді сітки з дванадцяти колонок яку ми будемо використовувати в нашій системі. Для пришвидшення роботи розробника ми будемо використовувати препроцесор SCSS. Він створює можливість робити стильову роботу над веб-додатком швидше та простіше.

Фреймворк React був обраний після проведення порівняльних характеристик з іншими фреймворками, які існують сьогодні та є популярні на сучасному ринку технологій для розробки клієнтської частини. Порівняння проводились серед звичайного JavaScript та двох версій Angular. Задача, яка була поставлена при порівнянні фреймворків – це зміна інформації в таблиці з даними в 10 000 рядках. За результатами проходження задачі було встановлено, що React має найбільшу швидкодію. Також, фрейморк має велику кількість методів, які спрямовані на оптимізацію роботи додатку.

Для серверної частини рейтингової системи використаємо програмну платформу мови програмування JavaScript Node.js. Вона надасть можливість використовувати обрану мову програмування для написання скриптів для серверної частини. Також, у нас є завдання створити односторінковий додаток (SPA), тому буде доречно використати фреймворк Express.js та на його базі створити API за яким ми будемо отримувати відповідні дані.

Для того щоб скомпонувати проект в єдине ціле та застосувати певні технології оптимізації використаємо Webpack. Даний білдер надає можливість провести мінімізацію HTML, CSS, JS файлів, оптимізацію зображень які, використовуються в проекті, створення спрайтів для іконок, які будуть використовуватися для забезпечення візуальної інтерпретації для певного функціоналу. Часто такий підхід підвищує зрозумілість інтерфейсу на інтуїтивному рівні. Оптимізація та мінімізація файлів зменшує розмір файлів, а спрайти зменшують кількість запитів до серверу. Мінімізація js файлів досягається за допомогою автоматизованого рефакторингу коду, який змінює назви змінних та видаляє усі переноси та пробіли в коді, створюючи суцільний блок коду. Читати та підтримувати такий код неймовірно важко, тому були створені *sourcemap* на період розробки системи. *Sourcemap* надають змогу повертати мінімізований код в початковий стан.

Для створення бази даних будемо використовувати MySql. Доступ до таблиць бази даних, модифікації та видалення існуючих рядків таблиць та інші задачі, які пов’язані з таблицями бази даних також будемо вирішувати за допомогою цієї мови запитів.

Розглянемо основну інформацію про технології та інструменти, які будуть використані при розробці веб-додатку.

1. HTML

HTML – стандартизована мова розмітки. Є основною мовою в мережі інтернет для відображення структури веб-сторінки. Актуальна версія цієї мови – 5. HTML документ складається з великої кількості елементів – тегів, які використовуються для відображення певних даних, форматування або службові (рис. 2.1). Можна розділити документ на декілька основних тегів – *!DOCTYPE, html, head, body*.

!DOCTYPE є службовим тегом і слугує для задання документу пояснення, що наступний код використовує останню версію HTML. Без вказання цього

тегу браузери можуть не зрозуміти теги, які були впроваджені лише в п’ятій версії мови.

Тег *html* не відображає нічого на сторінці. Він є корневим елементом і зберігає в собі всю майбутню структуру сайту та службовий блок *head*. Для звернення через мову CSS до цього тегу можна використати селектор *:root*. *Head* та *body* належать до блоку *html* та відповідають за наповнення сторінки.

Head – це службовий тег, в якому вказуються службові інструкції для браузеру, такі як задання стандарту кодування, опису та ключових слів тематики сайту, підключення файлів стилю та скриптів. Ключові слова та опис сайту задають відображення сайту в різних пошукових системах та SEO оптимізують сайт. Одним з способів задати певні налаштування – це передати їх у вигляді атрибутів тегу meta. Наприклад, для того щоб задати стандарт кодування необхідно додати до тегу *head <meta charset=”utf-8”>*.

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset="utf-8">
  <title>Моя первая веб-страница</title>
 </head>
 <body>
  <h1>Заголовок страницы</h1>
  <p>Основной текст.</p>
 </body>
</html>
```

Рисунок 2.1 Базова структура HTML документу

Тег body відображається у браузері та є контейнером, у якому знаходиться уся розмітка веб-сторінки. В HTML5 була створене поняття семантика за рахунок нових структурних елементів. Раніше, при створенні веб-сторінок розробники надавали блокам певне значення та логіку за допомогою додавання та відповідного найменування класів до елементів. Стилі найменування блоків в класах у розробників різні. Для створення єдиного рішення щодо семантики були створені відповідні теги (*header, nav, main, footer* та ін.). Пошукові роботи орієнтуються на ці теги та виконують певну

оптимізаційну роботу за аналізом та структурою цих тегів. Семантика спростила СЕО оптимізацію та процеси, які пов'язані з просуванням веб-додатків та сайтів у пошукових системах.

2. CSS та препроцесор SCSS

CSS – каскадні таблиці стилів, задають візуальну складову сайту. Файл підключається через тег *link* в тег *head*. Кожен тег має атрибути – певні параметри, які конкретизують роботу тегу та можуть передавати певну інформацію до елементу. Для того щоб звернутися до елементу треба скористатись одним з селекторів: ідентифікатор, клас, атрибут або тег. Ідентифікатор – це спеціальне значення елементу, яке має бути одним на весь документ, задається в документі CSS зі знаком # (рис. 2.2). Серед інших селекторів має найвищу специфіку – це означає, що властивості описані в даному селекторі будуть використовуватись в першу чергу та перекривати інші властивості.

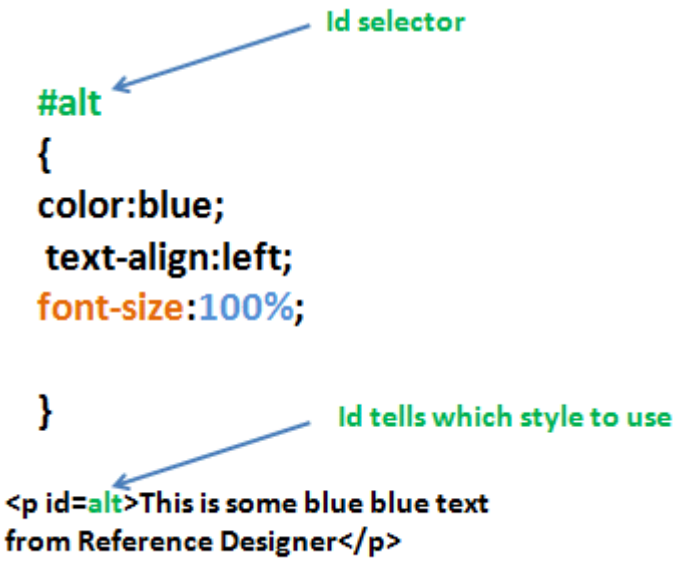


Рисунок 2.2 – Приклад використання ідентифікаторів

Клас задається за допомогою спеціального знаку «.» перед ім'ям, використовується для створення певного набору властивостей, які будуть використовуватись багатьма схожими елементами (рис. 2.3).

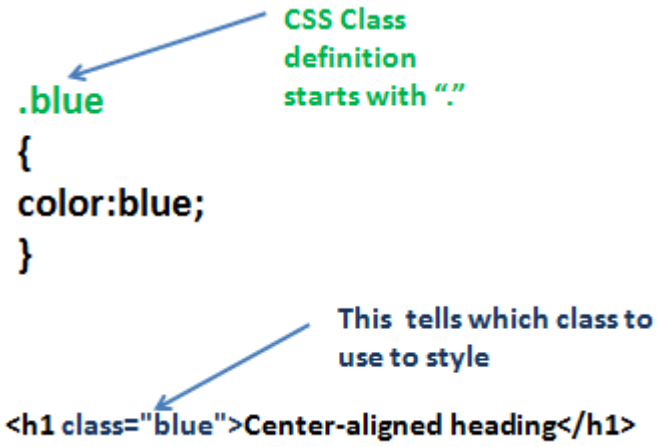


Рисунок 2.3 – Приклад використання класів

В нашому додатку буде використовуватись BEM методологія (рис 2.4.). Суть цієї методології в наступному, кожен елемент документу поділяється на блок, елемент та модифікатор. Блоком вважається незалежний елемент, який можна винести на іншу сторінку і він не втратить свою задумку. Усі елементи всередині блоку вважаються його елементами. Модифікатори – це блоки або елементи, які змінюють відображення селектору таким способом створюючи його модифікацію. Дана методологія має наступний запис *«.block__element--modificator»*. Ця методологія використовує тільки класи для доступу до елементів. Попри можливу завантаженість та нечитабельність коду ця методика дає більше переваг ніж недоліків.

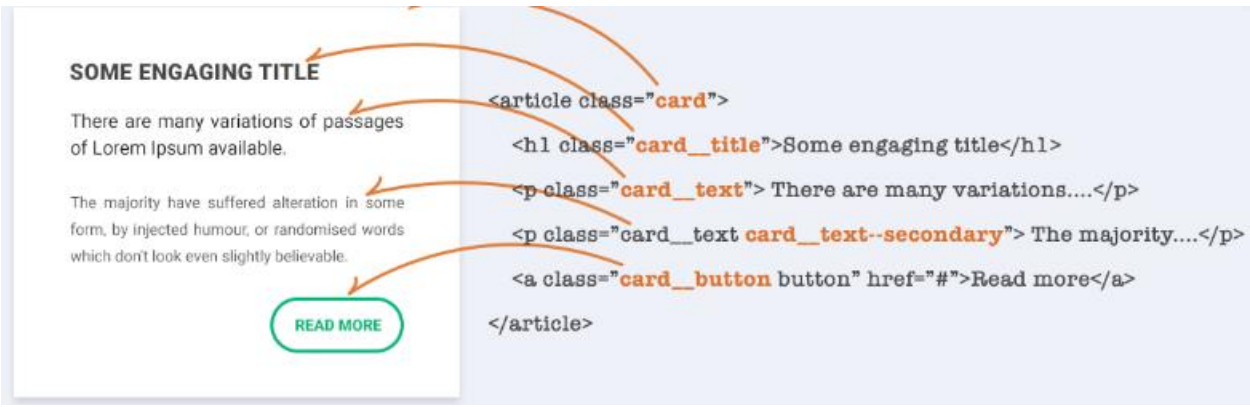


Рисунок 2.4 – Приклад використання ідентифікаторів

SCSS – це препроцесор, який спрощує роботу з CSS та мінімізує його недоліки. Основними особливостями препроцесору є можливість створення вкладеності селекторів, створення змінних, імітація циклів, міксінів та

розширень, які допомагають використовувати велику кількість разів блоки властивостей для інших елементів, розділення документу на частини не впливаючи на швидкодію роботи під час http запитів. Розглянемо ці особливості детальніше.

Перша особливість, яку ми розглянемо – імпорт (рис. 2.5). CSS має свій функціонал для імпорту (виконується за допомогою аналогічної функції *@import*). Різниця цих способів в швидкодії. Оригінальний функціонал *@import* у CSS при кожній ін'єкції файлів створює новий http запит. На практиці для створення зручної структури файлів мова йде про десятки імпортів. *@import* в SCSS виправив цей недолік і взяв лише переваги такого способу.

Рисунок 2.5 – Приклад роботи *@import* в SCSS

Дуже спрощує роботу розробника така особливість як наявність змінних(рис. 2.6). Змінні в препроцесорі SCSS можуть зберігати будь-які інформацію – від назви css властивостей до певних рядків тексту, які можуть повторюватись в коді. Часто в проектах створюють окремий файл з змінними, які зберігають кольори, текст, значення властивостей.

Рисунок 2.6 – Приклад застосування змінних в SCSS

Для можливості використання набору властивостей з певним значенням в різних частинах коду можна застосувати міксини (рис. 2.7). Часто застосовують для задання певної властивості з врахуванням усіх вендорних префіксів. Це гарантує, що вендорні префікси не забудуться розробником і будуть гарантовано прописані в коді. Інша ситуація в якій використовують міксини – це створення модульності стилів. За допомогою міксину створюється набір властивостей для певного блоку, який можна модифікувати за допомогою вхідних параметрів і аналогічним чином застосовується в потрібних частинах коду.

```scss
@mixin transform($property) {
  -webkit-transform: $property;
  -ms-transform: $property;
  transform: $property;
}
.box { @include transform(rotate(30deg)); }
```

```scss
.box {
  -webkit-transform: rotate(30deg);
  -ms-transform: rotate(30deg);
  transform: rotate(30deg);
}
```

Рисунок 2.7 – Приклад застосування міксинів в SCSS

Схожий функціонал мають розширення(рис. 2.8). Розширення – це можливість використовувати класи знову і знову за допомогою спеціального функціоналу препроцесора. На практиці існує такий підхід для створення модифікацій: створюється клас з властивостями (властивості повинні бути ті, які повторюються в кожній з модифікацій (блочні особливості, розміри і т.д)), далі створюється клас з модифікацією та за допомогою *@extend* додається клас з базовими властивостями. Також є можливість створювати класи, які будуть наслідуватися та не будуть відображені в результуючому файлі CSS якщо не використовувались за допомогою знаку % перед класом.

```scss
/* This CSS will print because %message-shared is extended. */
%message-shared {
  border: 1px solid #ccc;
  padding: 10px;
  color: #333;
}

// This CSS won't print because %equal-heights is never extended.
%equal-heights {
  display: flex;
  flex-wrap: wrap;
}

.message {
  @extend %message-shared;
}

.success {
  @extend %message-shared;
  border-color: green;
}

.error {
  @extend %message-shared;
  border-color: red;
}

.warning {
  @extend %message-shared;
  border-color: yellow;
}
```

```scss
.message, .success, .error, .warning {
  border: 1px solid #ccc;
  padding: 10px;
  color: #333;
}

.success {
  border-color: green;
}

.error {
  border-color: red;
}

.warning {
  border-color: yellow;
}
```

Рисунок 2.8 – Приклад застосування розширень в SCSS

SCSS має можливість створювати вкладеність властивостей. Це спрощує читабельність та розуміння коду. Проте бувають ситуації, коли необхідно вивести певні властивості в корінь, зберігши при цьому розуміння до якого блоку властивостей цей код належить. Це можна виконати за допомогою *@at-root* (рис. 2.9). Працює даний функціонал для вкладеностей першого рівня.

```scss
.parent {
  ...
  @at-root {
    .child1 { ... }
    .child2 { ... }
  }
  .step-child { ... }
}
```

```scss
.parent { ... }
.child1 { ... }
.child2 { ... }
.parent .step-child { ... }
```

Рисунок 2.9 – Приклад застосування *@at-root* в SCSS

3. Фреймворк Bootstrap та його React бібліотека

Bootstrap – це css фреймворк, який надає велику кількість готових компонентів з можливістю подальшої зміни під потреби користувачів або розробників. Можна розділити на декілька частин – це сітка для розмітки сторінки та саме готові компоненти (рис. 2.10). На сьогоднішній день актуальна версія фреймворку Bootstrap четверта.

Сітка представляє собою розмітку з дванадцяти колонок, яка вираховується у відсотках відповідно до розміру екрану девайсу. Основними структурними класами, які відповідають за побудову сітки є .container, .row, .col. Кожен з цих класів має свої показники внутрішніх та зовнішніх відступів (наприклад, у .row від'ємні зовнішні відступи, а .col за замовчуванням має внутрішні відступи величиною відповідною до величини зовнішніх відступів .row). Тому для створення сітки, яка матиме прогнозовану поведінку з точки зору стилів необхідно щоб в класі .row вкладеним елементом був блок з класом .col.

Клас .row виступає структурним класом для формування рядків з контентом. В четвертій версії фреймворку Bootstrap клас використовує flex технологію для розміщення елементів. На даний момент flex однозначно є найбільш популярним та найбільш зручним способом створення розмітки. Перевагою цієї технології є можливість вирівнювати внутрішні елементи як по горизонталі, так і по вертикалі.

За допомогою класів .col та їх похідних можна задати розмір блоку. Щоб вказати скільки колонок займатиме блок по сітці необхідно блоку задати певне значення від 1 до 12 класу .col-n (n – значення розміру). При переповненні сітки блоки зміщуються на новий рядок і продовжують заповнювати документ. Для впливу на розмір контейнеру в якому будуть розміщені колонки необхідно використовувати класи .container та .container-fluid. .container

займає певний фіксований розмір документу і не змінюється при збільшенні розміру екрану, .container-fluid розтягується на всю ширину документу.

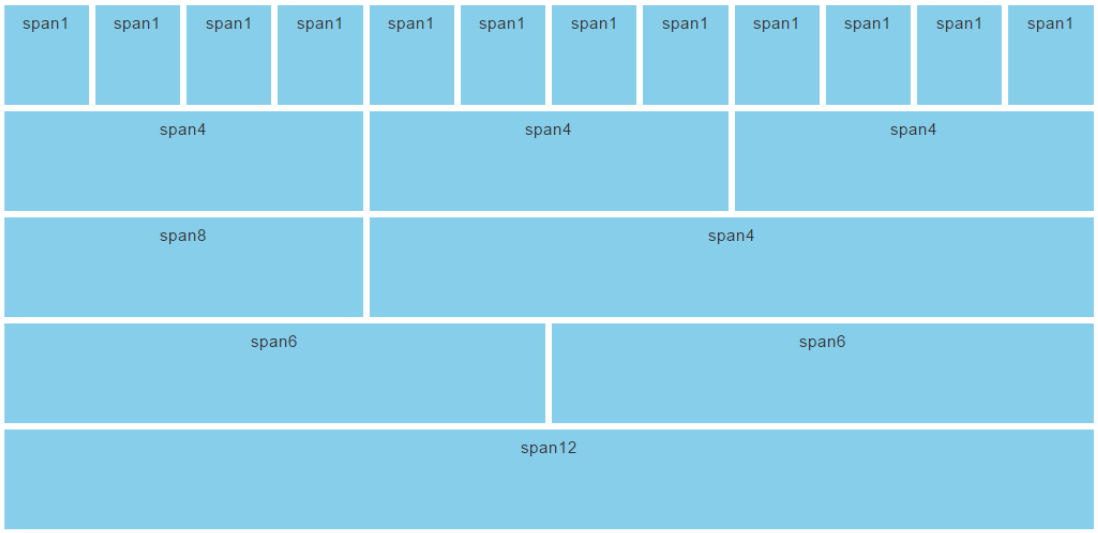


Рисунок 2.10 – Приклад сіточної системи фреймворку Bootstrap

Фреймворк Bootstrap має вбудовану адаптивність для класів сітки. Таким чином можна за допомогою відповідних класів створювати коректну верстку на всіх девайсах. Для того щоб задати розмір блоку для певної роздільної здатності необхідно використовувати спеціальні суфікси xs, sm, md, lg, xl. Ці суфікси відповідають за роздільну здатність менше 576 пікселів, від 576 до 768 пікселів, від 768 до 992 пікселів, від 992 до 1199 пікселів та більше 1200 пікселів відповідно. Наприклад, для того, щоб задати блоку розмір 6 для роздільної здатності телефонів (менше 576 пікселів) використовуємо клас .col-xs-6.

Так як ми використовуємо технологію SCSS в нашій роботі, то можемо скористатися міксинами Bootstrap для створення додаткової адаптивності для різних властивостей (рис. 2.11). Міксини, які дуже часто використовуються це media-breakpoint-down, media-breakpoint-between та media-breakpoint-only. Вони є готовими медіа запитами, які використовуються для задання певних стильових правил до певного розміру екрану та менше, до певного проміжку роздільної здатності (наприклад, від 768px до 1024px) та для створення правил для єдиного значення роздільної здатності відповідно.

```scss
@include media-breakpoint-down(xs) { ... }
@include media-breakpoint-down(sm) { ... }
@include media-breakpoint-down(md) { ... }
@include media-breakpoint-down(lg) { ... }
// No media query necessary for xl breakpoint as it has no upper bound on its width

// Example: Style from medium breakpoint and down
@include media-breakpoint-down(md) {
  .custom-class {
    display: block;
  }
}
```

Рисунок 2.11 – Міксини для адаптивності в Bootstrap

Для розробників, які використовують React в розробці проекту була створена бібліотека, яка реалізує готові елементи та інші рішення у вигляді компонентів. Наприклад, блоки з класом container, які слугували найбільшим структурним блоком, який зберігає в собі всі інші елементи був замінений на компонент Container (рис. 2.12). Блоки з класами row та col (та його похідні класи) були замінені на Row та Col компоненти відповідно. Також, ця бібліотека має багато готових стилізованих компонентів. Одним з таких є форма відправлення запитів на сервер. Компонент має зрозумілу назву <Form> та має наступні елементи форм: <FormGroup> для групування полів введення за певною семантикою, <FormControl>, який реалізує поле вводу даних та <FormLabel> для створення підписів до полів вводу інформації.

```jsx
<Container>
  <Row>
    <Col>1 of 3</Col>
    <Col xs={6}>2 of 3 (wider)</Col>
    <Col>3 of 3</Col>
  </Row>
  <Row>
    <Col>1 of 3</Col>
    <Col xs={5}>2 of 3 (wider)</Col>
    <Col>3 of 3</Col>
  </Row>
</Container>
```

Рисунок 2.12 – Приклад розмітки з залученням компонентів бібліотеки react-bootstrap

4. JS

JS – це функціональна мова розробки веб-додатків, основна мова, яка використовується для створення сценаріїв та логіки для сайтів та інших можливих видів веб-додатків. Основні задачі для яких використовується це маніпуляції з елементами веб-сторінки, реакція на події комп'ютерної миші, клавіатури. Особливості мови JavaScript, які ми розглянемо в цьому розділі це нетипізованість мови, типи даних, експорт та імпорт модулів, асинхронність, DOM-дерево, інтерфейсні події.

Особливість цієї мови, яка виділяє цю мову серед інших це нетипізованість змінних. Це означає, що при оголошенні змінних ми не повинні задавати тип даних. Перевага такої особливості в тому, що ми можемо використовувати одну змінну для різних даних і на виході ми не побачимо помилок. Недоліком є те, що на певних етапах та в певних функціях є необхідним робота з певним типом даних. А завдяки нетипізованості мови та її внутрішнім технологіям таким як преведення типів, контролювати дані стає складно.

JavaScript налічує 7 типів даних: number, string, boolean, null, undefined, object та symbol.

- Number створений для роботи з будь-якими числами (цілі та з плаваючою крапкою),
- String – для рядків та текстових символів,
- Boolean – логічний тип, має значення true та false,
- Null – окремий тип, використовується для невідомих значень,
- Undefined – окремий тип, використовується для значень, які не були задані,
- Object – складний тип даних, за допомогою цього типу існують більш складні структури даних (функції, масиви, об'єкти),
- Symbol – тип даних, який створює унікальні ідентифікатори.

Експорт та імпорт даних надає можливість розділяти код на різні файли і таким чином структурувати код та полегшувати його читабельність (рис 2.13). За допомогою службового слова *export* ми даємо знати, що цей модуль буде використовуватись в інших файлах. Для того щоб додати модуль в інший документ необхідно використати інше службове слово *import*, вказати ім'я, за яким цей елемент буде використовуватись в цьому файлі та вказати шлях до модуля за допомогою *from 'path'*. Є випадки коли один файл експортує декілька функціональних елементів. В такому випадку для імпорту необхідно написати *import { element1, element2 } from 'path'*.

```
import React from 'react';
import PropTypes from 'prop-types';

import './apple.scss';

function Apple(props) {
  const { position } = props;
  return (
    <div className="apple" style={position} />
  );
}

Apple.propTypes = {
  position: PropTypes.objectOf(PropTypes.string).isRequired,
};

export default Apple;
```

Рисунок 2.13 – Приклад використовування імпортів та експортів при написанні компонентів

Асинхронність – це можливість відкласти виконання певної частини коду на певний час або продовжити виконання наступного коду не затримуючи і не перевантажуючи систему виконанням коду.

Document Object Model або DOM-дерево – це представлення HTML документу у вигляді дерева тегів. Корневим або батьківським елементом усіх вкладених елементів є тег *<html>*. Таким чином, маючи доступ к об'єктам DOM-дерева ми можемо маніпулювати HTML сторінкою (рис. 2.14).

```
document.body.style.background = 'red';
```

Рисунок 2.14 – Приклад стилізації тега *<body>*

За допомогою інтерфейсних подій можна додавати певний інтерактив на сторінку веб-додатку. Найбільш популярними подіями серед подій миші є

- click – виконується після натиснення на ліву кнопку миші,
- mouseover – виконується після наведення на певний елемент сторінки,
- mouseup – виконується після натиснення кнопки миші,
- mousedown – виконується після того як кнопка миші була відпущена,

Найбільш популярними подіями серед подій клавіатури є keydown та keyup – відбувається після натиснення та після того як клавіша була відпущена відповідно.

5. React

React.js – це JavaScript бібліотека для розробки інтерфейсу користувачів. Веб-сторінка ділиться на велику кількість компонентів. Усі компоненти можна поділити на компоненти двох типів: stateless та stateful. Stateless компоненти це компоненти, які не мають певної логіки. Тобто, це просто HTML розмітка компоненту. Statefull компоненти це компоненти які мають певний функціонал, реалізацію певних життєвих циклів компоненту, state об’єкт.

Особливостями даної бібліотеки є однонаправлена передача даних, використання віртуального DOM, JSX.

JSX – розширення мови JavaScript. Він допускає можливість створювати HTML розмітку в файлі JS та проводити певну роботу над цією розміткою.

Кожен компонент має свій життєвий цикл (рис. 2.15). Усі цикли починаються з прилаштування компоненту (constructor, render – головний та обов’язковий метод кожного компоненту, componentDidMount методи), можуть змінюватись в ході своєї роботи (componentDidUpdate методи) та завершує свій цикл роботи (componentWillUnmount метод). Також є можливість контролювати оновлення компонента за допомогою методу

shouldComponentUpdate. Головне завдання цього методу - вплив на швидкодію додатку.

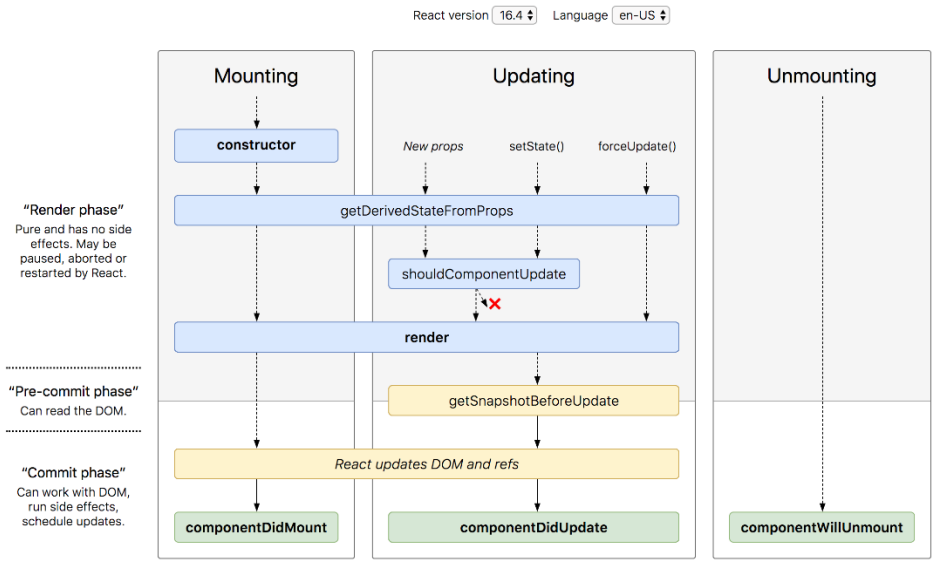


Рисунок 2.15 – Методи життєвого циклу компоненту

6. Node.js, Express.js

Express.js – це фреймворк для веб-додатків Node.js. За допомогою цієї технології ми будемо створювати серверну частину веб-додатку. В рамках розробки серверної сторони необхідною частиною є розробка маршрутизації. Маршрутизація – це технологія, яка відповідає за коректну відповідь на клієнтські запити та відповідні методи запитів HTTP. Для реалізації функціоналу додавання, відображення та зміни інформації необхідно на серверній стороні реалізувати методи запитів та визначити певні URI для яких ця логіка буде працювати. Серед них є GET, POST, DELETE методи запити.

GET відповідає за надання певних даних за відповідним URI (рис 2.16). За допомогою цього методу ми можемо тільки отримувати дані, видаленню та зміні вони не піддаються.

```
app.get('/', function (req, res) {
  res.send('Hello World!');
});
```

Рисунок 2.16 – Реалізація get методу

POST відповідає за отримання певної інформації на клієнтській частині та відправка цих даних на сервер з метою подальшою зміною даних в базі даних (рис. 2.17).

```
app.post('/', function (req, res) {
  res.send('Got a POST request');
});
```

Рисунок 2.17 – Реалізація post методу

DELETE відповідає за отримання певної інформації на клієнтській частині та відправка запиту на видалення даних на сервер (рис. 2.18).

```
app.delete('/user', function (req, res) {
  res.send('Got a DELETE request at /user');
});
```

Рисунок 2.18 – Реалізація delete методу

7. Webpack

Webpack – це статичний білдер модулів для сучасних JavaScript веб-додатків. За допомогою файлу конфігурації білдеру є можливість налаштувати відповідно до потреб нашого веб-додатку (додаток А).

Суть роботи цього інструменту в наступному. Усі файли, які створюються під час розробки це HTML, CSS, JS файли, файли зображень, шрифтів об’єднуються в один файл з метою зменшення кількості HTTP запитів та відповідно зменшення швидкості завантаження файлів веб-сторінкою (рис. 2.19). За допомогою підключення плагінів та лоадерів можемо реалізувати дані можливості.

Єдиним файлом слугує js файл. За допомогою конфігураційного файлу ми задаємо файл, який буде слугувати вхідним файлом (Entry властивість), вихідний файл та шлях до нього (output властивість). Окрім js файлів, в проекті використовуються файли зображень, шрифтів, стилів. Для того, щоб їх коректно налаштувати та отримати необхідний нам результат використовуємо лоадери. В даному проекті використовуються лоадер для трансформації js

коду для коректної роботи та розуміння в найпопулярніших браузерах. Для правильного написання коду використаємо eslint-loader. Він використовує певний набір правил та практик для створення максимально підтримуваного та простого для розуміння коду та використовується перед трансформацією коду.

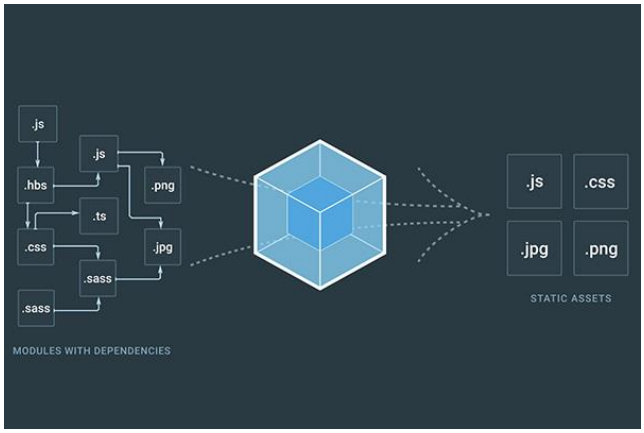


Рис. 2.19 – суть роботи Webpack

Детальніше про плагіни, які використовуються у конфігураційному файлі. MiniCssExtractPlugin використовується для відокремлення CSS файлу з усього скомбінованого файлу. Єдиний файл використовується під час розробки та швидкого локального розгортування проекту, проте серверу необхідні файли HTML, CSS, JS файли. Після відокремлення файл мінімізується та створюються соурсмапи, тому ми використовуємо цей плагін. HtmlWebpackPlugin використовуємо для генерації Html файлу з підключенням необхідних css та js файлів. CleanWebpackPlugin використовуємо для очищення тимчасових файлів підчас розробки. SvgSpritePlugin використовуємо для генерації спрайтів svg файлів.

Висновки до розділу

В цьому розділі були проаналізовані та описані технології та інструменти розробки інформаційних систем, які існують на ринку. Для розробки клієнтської частини були обрані HTML, CSS та JavaScript. Версія мови розмітки та каскадних таблиць стилів були актуальні – 5 та 3 відповідно.

JavaScript код був написаний орієнтуючись на останні стандартів ECMAScript ES6+. Для написання стилів був задіяний фреймворк Bootstrap та його реалізація для React react-bootstrap. Для створення єдиної логіки найменування класів застосовувалась методологія BEM. Особливість цієї методології в підході до найменування елементів розмітки. Кожен елемент на розгляд розробника поділяється на блок, елемент або модифікатор. Для пришвидшення написання коду та створення зручних для читання файлів стилів був задіяний препроцесор SCSS. Цей інструмент надає можливість створювати вкладеність стильових правил, використовувати певні блоки коду в інших правилах за допомогою міксинів та екстендів, створювати змінні та застосовувати їх в різних правилах стилів. Для пришвидшення роботи клієнтської частини вона була реалізована за допомогою бібліотеки React. Ця технологія була обрана після аналізу сучасних фреймворків. Для порівняння було обрано VanillaJS, Angular та React. Було виявлено, що бібліотека React найшвидше справляється з відображенням великої кількості даних та рендером DOM-дерева краще за конкурентів. Ця бібліотека має особливий підхід до рендерингу елементів на сторінці, який відрізняється від звичайного рендерингу DOM-дерева. Використовуючи так званий віртуальний DOM та механізм узгодження (пов’язаний з взаємодією з DOM за допомогою допоміжної бібліотеки ReactDOM), компонентний підхід до розробки та події життевого циклу компонентів була оптимізована робота клієнтської частини рейтингової системи.

РОЗДІЛ 3. РОЗРОБКА СИСТЕМИ ДЛЯ НАВЧАННЯ СТУДЕНТІВ ТА КЕРУВАННЯ НАВЧАЛЬНИМ ПРОЦЕСОМ

3.1. Архітектура системи

Рейтингова система навчання студентів буде складатися з декількох модулів (рис.3.1). Серед них є модуль результатів, який відображає результати студентів відсортовані за вихідним значенням, модуль завдань, які виконувались студентами, модуль формування завдань для студентів, модуль новин та оголошень та модуль авторизації користувачів системи.

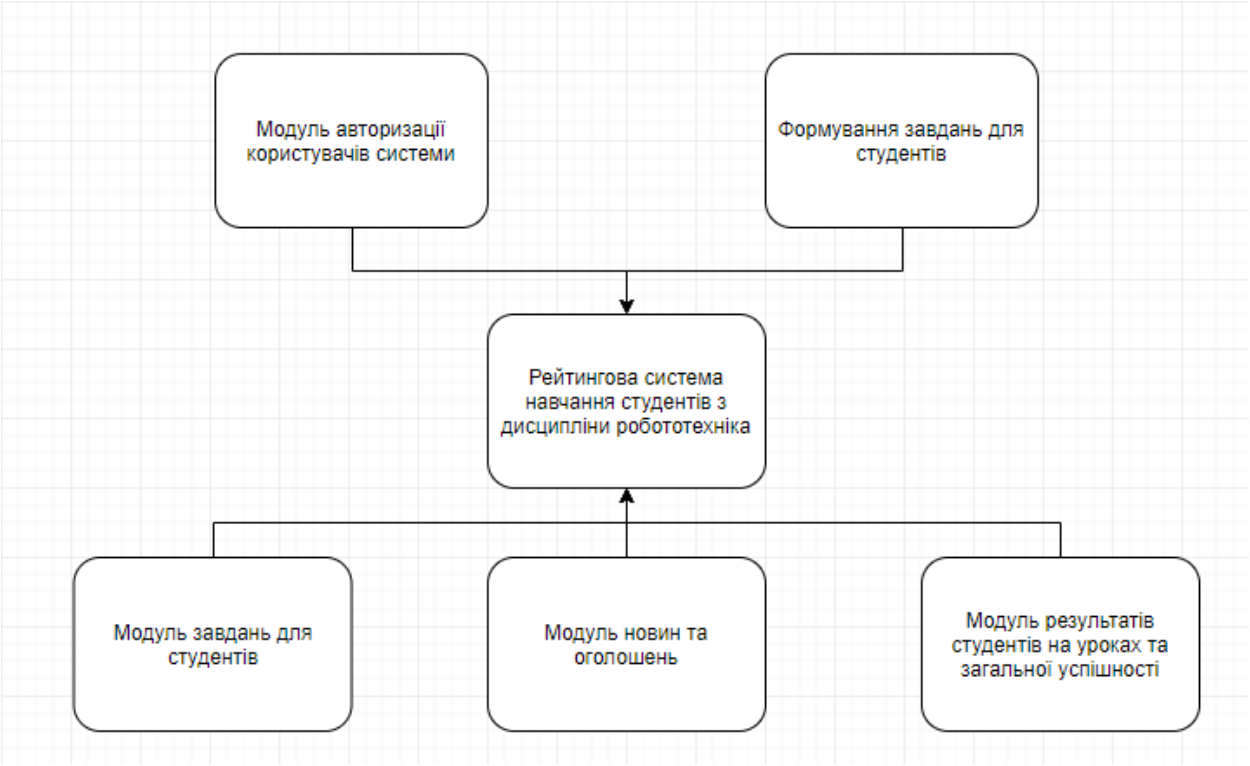


Рисунок 3.1. Архітектура системи для навчання студентів

Першим розберемо модуль з якого буде починатися робота усіх користувачів – це модуль авторизації користувачів системи. Один з основних модулів системи, забезпечує захист даних користувачів від зовнішнього розповсюдження та розділення інформації для створення кастомізованого підходу до кожного студенту. Для того щоб зайти в рейтингову систему необхідно зареєструватися. Для регістрації необхідно ввести мінімальні дані: електронну пошту та пароль. Електронна пошта виступає логіном для даного

користувача та буде задіяна системою для підтвердження певних дій необхідних для успішної валідації користувача та для розсилки певного контенту від компанії. Для забезпечення можливості швидко зареєструватися створена кнопка, яка реєструє користувача в системі за допомогою існуючого акаунту в gmail.com. Для випадків, коли користувач забув або не знає пароль додана кнопка «Нагадати пароль». В цій ситуації користувачу необхідно ввести свою електронну пошту, яка була зареєстрована в цій системі раніше, на цю пошту буде відправлений лист з посиланням на відновлення доступу до особистого кабінету. Валідація полів вводу пошти та паролю виглядає наступним чином. Не знаючи усі можливі почтові агенти, ми можемо орієнтуватися тільки на установлену структуру запису пошти. Тому приблизним шаблоном для коректної електронної пошти буде *@*.*. Важливішим є валідація паролю до акаунту користувача. Основні вимоги це: розмір паролю (більше 5 символів), мова (пароль повинен бути написаний на латиниці), та наявність спеціальних символів (*,%,-,_, як мінімум один символ).

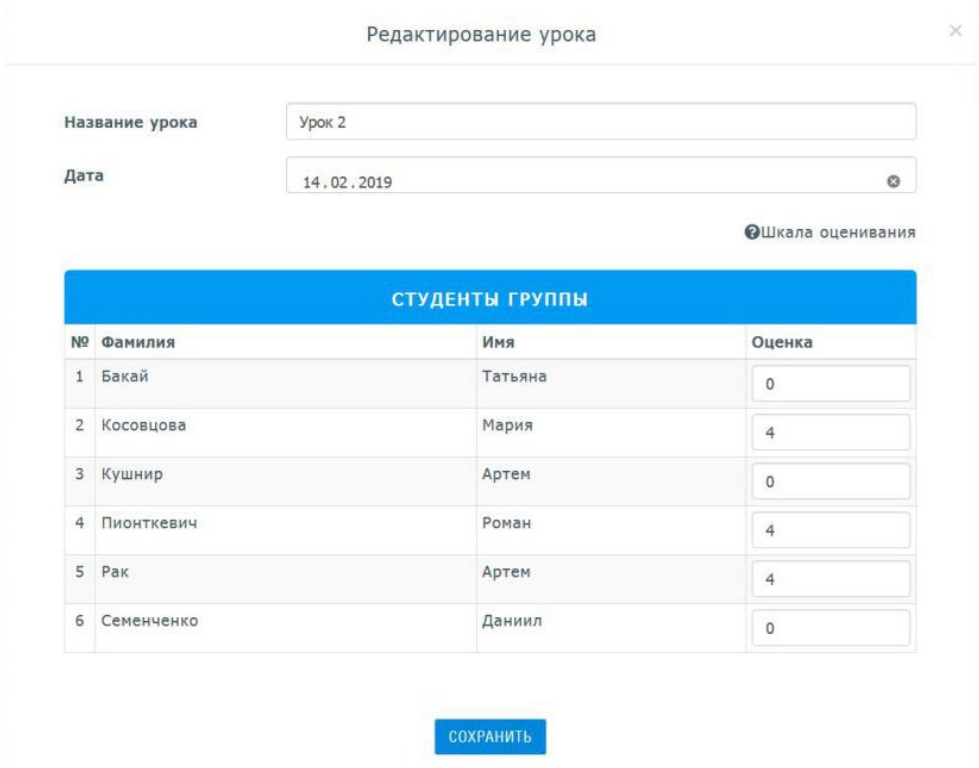


Рисунок 3.2. Інтерфейс для редагування групи студентів та їх успішності

Користувачів за кількістю прав можна поділити на 3 категорії: студенти, викладачі та адміністратори. Адміністратори мають найбільший спектр прав, можуть створювати студентів та викладачів, корегувати інформацію про студентів в рейтинговій системі та створювати та редагувати новини. Студенти мають read-only доступ до всієї інформації (новини, завдання, результати). Викладачі мають аналогічний набір прав зі студентами за виключенням можливості заповнювати систему успішності студентів.

Зайшовши в систему, користувач бачить вкладку з новинами, вкладку з налаштуваннями для корекції власної інформації та зміни паролю, вкладку з існуючими завданнями з певної тематики та власне рейтингову систему. За замовчуванням відкрита вкладка з новинами. Новини створюються адміністратором та іншими користувачами, яким були видані адміністраторські права. Навігація знаходиться у лівій частині екрану, в зручному місці для користувача. Навігація складається з таких категорій: Новини, Задачі, Мій рейтинг та Мій профіль.

У вкладці з завданнями розміщені усі задачі для студентів. Задачі зображені у вигляді таблиці. Для того щоб детально ознайомитись з новиною необхідно натиснути на новину. Під обраною новиною з’явиться додатковий блок з детальним описом задачі. До опису належить назва задачі, короткий опис, кількість балів за виконання цього завдання та складність, яка представлена у вигляді шкали від 1 до 5. Після вибору завдання можна виконати його клікнувши по кнопці «Обрати завдання». Додані задачі відображаються у вкладці з задачами. У вкладці Мої задачі виконані задачі відрізняються від звичайних. Виконані задачі виконані в сірих тонах імітуючи неактивний стан задачі.

Для роботи з завданнями для студентів був створений модуль формування завдань. Основні задачі, які виконує даний модуль це створення нових завдань, редагування та видалення вже існуючих завдань. Для створення нового завдання необхідно заповнити поля вводу інформації про

заголовок завдання, його опис, вказати складність завдання та опублікувати його для того, щоб воно відображувалось на сторінці завдань. Завдання може також супроводжуватись зображеннями, іншими графічними елементами та посиланнями, які ведуть на корисну інформацію. Створювати, модифікувати та видаляти завдання можуть викладачі та адміністратори.

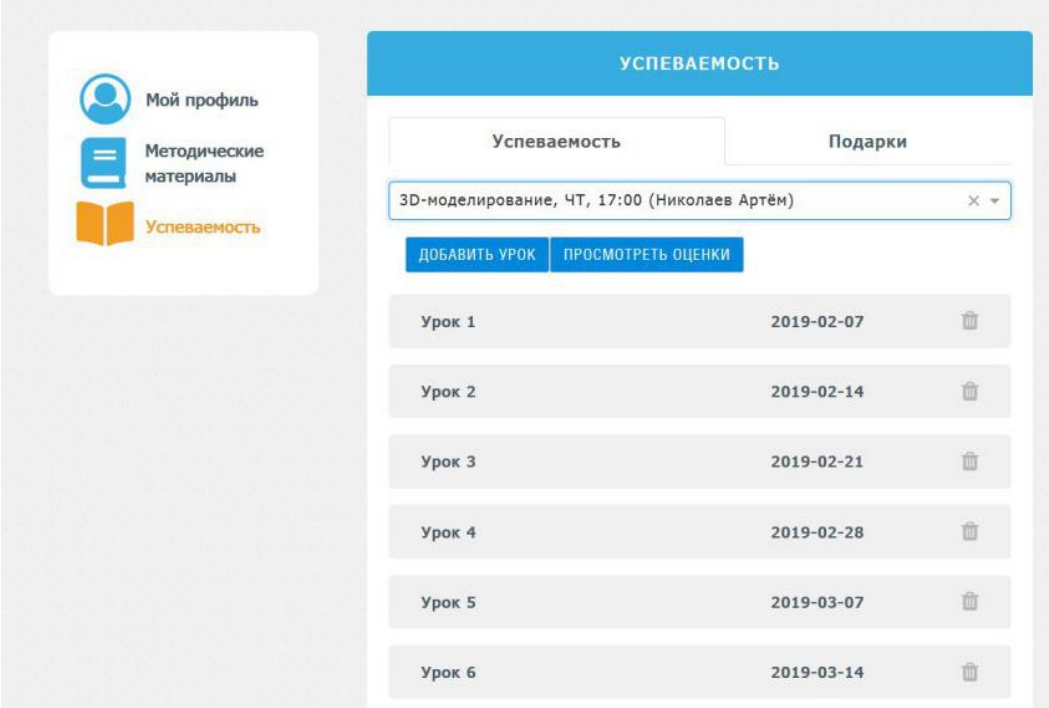


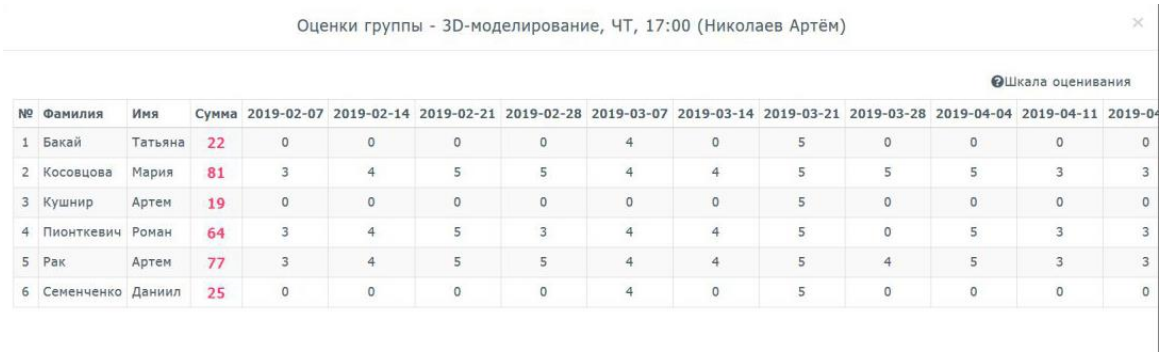
Рисунок 3.4. Інтерфейс особистого кабінету рейтингової системи

Для студентів був створений аналогічний модуль завдань, який відображує усі завдання, створені адміністратором ресурсу або викладачем та які були опубліковані для можливості публічного доступу до них. Для зручності пошуку завдань існує поле пошуку завдань за ім’ям, створена сортування за алфавітом за зростанням або за спаданням, додана фільтрація за рівнем складності. При великій кількості завдань завдання розбиваються на групи по 20 завдань. З’являється пагінація з нумерацією сторінок та стрілками для переходу на одну сторінку вперед або назад. При великій кількості сторінок відображається обмежена кількість сторінок, інші заміняються символом «…». Таким чином, навігація у випадку великої кількості сторінок складається з першої сторінки, сторінки до активної, поточна сторінка,

наступна сторінка від поточної та остання сторінка. Для підвищення швидкості відображення завдань, запити до серверу за завданнями поділені на блоки по 20 завдань.

Для інформування користувачів веб-орієнтованої системи про існуючі акції, передачі інформації, яка пов’язана з навчальним курсом студентів та для ознайомлення студентів з новинами, пов’язаними з напрямом навчання та схожою тематикою був створений модуль новин та оголошень. Новини будуть займати всю сторінку та у випадку великої кількості новин буде з’являтись пагінація. На одній сторінці можуть відображатися 10 новин. Структура однієї новини наступна: зображення, яке пов’язане з тематикою новини, заголовок новини, дата публікації та короткий опис новини (відображення опису новини на сторінці усіх новин регулюється при створенні). Для внесення додаткового розуміння користувачам новини та акційні пропозиції будуть додатково помічатися спеціальними бейджами з видом новини (акційна пропозиція, оголошення чи звичайна новина), які одразу інформують студента.

Основний модуль, який став ціллю цієї роботи це модуль результатів студентів на уроках та загальної успішності(рис.3.5).


Оценки группы - 3D-моделирование, ЧТ, 17:00 (Николаев Артём)

Шкала оценивания

№	Фамилия	Имя	Сумма	2019-02-07	2019-02-14	2019-02-21	2019-02-28	2019-03-07	2019-03-14	2019-03-21	2019-03-28	2019-04-04	2019-04-11	2019-04
1	Бакай	Татьяна	22	0	0	0	0	4	0	5	0	0	0	0
2	Косовцова	Мария	81	3	4	5	5	4	4	5	5	5	3	3
3	Кушнир	Артем	19	0	0	0	0	0	0	5	0	0	0	0
4	Пионткевич	Роман	64	3	4	5	3	4	4	5	0	5	3	3
5	Рак	Артем	77	3	4	5	5	4	4	5	4	5	3	3
6	Семенченко	Даниил	25	0	0	0	0	4	0	5	0	0	0	0

Рисунок 3.5. Макет графічної частини модулю рейтингової системи

В особистому кабінеті виділена окрема вкладка для цієї сторінки. При переході на цю вкладку відображається в правій частині екрану рейтингова система навчання. Для кожного користувача буде відображатися рейтингова система групи, в якій він навчається. Вона представлена певною кількістю людей, які навчаються в цій групі та загальним рейтингом з певного напряму

навчання для кожного студента. Рейтингова система оформлена у вигляді таблиці. Список людей може сортуватися за алфавітним порядком або за величиною рейтингу студента. Натискаючи на рядок певного користувача висувається додатковий блок, в якому знаходяться виконані завдання з більш детальним описом завдання. Також в цьому блоці можна побачити детальну інформацію про отримані бали за виконані завдання.

3.2. Структура бази даних

Для нашої задачі необхідно створити сховище для збереження даних від користувачів, дані про задачі, створені групи та рейтинг студентів у певній групі. Для цього необхідно визначити структуру бази даних. Після аналізу даних, які будуть використовуватись у додатку було прийнято рішення створити 9 таблиць в базі даних. Були створені таблиці для користувачів, груп, завдань, новин, рейтингових записів та декілька допоміжних таблиць для створення зв'язку між таблицями. Таблиці нашої бази даних мають префікс lc. Для створення унікальних імен таблиць та можливості інтегрувати нашу систему в інші з метою масштабування необхідно було задати унікальний префікс.

Структура бази даних для рейтингової системи навчання студентів має наступний вигляд(рис.3.5).

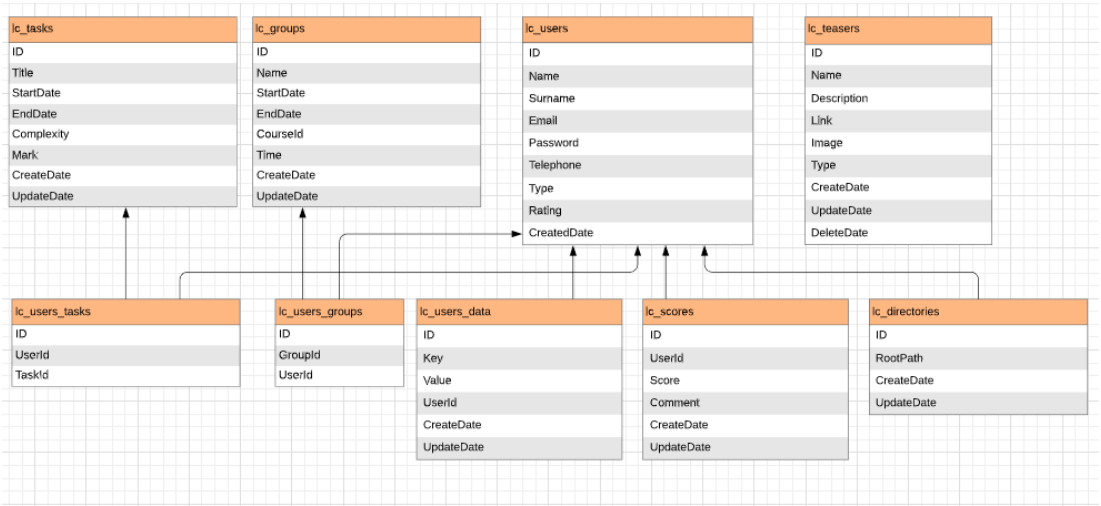


Рисунок 3.5. Структура бази даних рейтингової системи

Для нашої бази даних було створено наступні таблиці: lc_users, lc_teasers, lc_groups, lc_users_group, lc_users_data, lc_scores, lc_directories, lc_tasks, lc_users_tasks. Lc_users_data, lc_users_tasks, lc_users_group створені для зв'язку між таблицями. Розглянемо детальніше призначення кожної з таблиць в структурі бази даних для рейтингової системи навчання.

Lc_groups – це таблиця в базі даних, яка зберігає інформацію про групи людей, поділених за тематикою та набором. Поля, які в нашій базі описують групи це назва, дата створення групи, можлива дата останнього оновлення таблиці, дата початку курсу та кінцева дата проходження курсу та часові рамки, в які проходять завдання. Для створення зв'язку з таблицею lc_users була згенерована проміжна таблиця lc_users_groups, яка зберігає унікальній ідентифікатор групи та користувача.

Таблиця lc_users створена для збереження особистої інформації про користувачів. Таблиця налічує наступні поля: унікальний ідентифікатор, ім'я користувача, прізвище, пароль від аккаунту, телефон, тип користувача, Його загальний рейтинговий бал та додаткові службові поля для інформування про дату створення користувача та дату оновлення даних.

Таблиця lc_users_data створена для опціональної інформації від користувача. Були передбачені ситуації, коли користувач вказує декілька номерів телефону або додаткові електронні пошти. Головна таблиця lc_users завантажує основну інформацію про користувача. Додаткова інформація завантажується при необхідності за допомогою додаткового запиту. Таким чином, ми маємо компактну таблицю користувачів lc_users з фіксованою величиною полів та додаткову таблицю lc_users_data кількість полів якої може варіюватися.

Для того щоб зберігати дані, які будуть використовуватися при створенні новин та оголошень була створена таблиця lc_teasers. Дана таблиця налічує наступні поля: назва новини (використовується як заголовок матеріалу), посилання (необов'язкове поле, яке дає можливість залишити

посилання на джерело інформації), тип (новина або оголошення (акційні пропозиції, набори в групу на певний курс і т.д.), опис новини (власне тіло новини, яке передає основну суть матеріалу), зображення новини, дата створення новини та дата останнього оновлення будь-якої інформації з цих полів.

Таблиця lc_scores зберігає дані про користувача, які залишає користувач після проходження курсу у вигляді коментарів. Не всі користувачі бажають залишати коментар, тому форма заповнення коментарів необов'язкова.

В таблиці lc_directories зберігається службова інформація про структуру директорій в нашому проекті. Для додаткової зручності можна створювати вкладеність директорій для кастомізації та створення зручної структури папок та файлів.

Потенційно найбільша таблиця за кількістю даних це lc_tasks. В ній зберігаються завдання для студентів усіх груп та усіх напрямів. Ця таблиця має наступні поля: унікальний ідентифікатор завдання, назва завдання, опис задачі, яку необхідно виконати, правильне вирішення цієї задачі, дата створення та дата модифікації задачі, відмітка складності задачі, оцінка, яку отримає користувач за успішне виконання задачі.

3.3. Налаштування проекту

Перший задача, яка стоїть перед розробкою додатку – це налаштування системи. Для кожного проекту створюються службові файли з налаштуваннями. Одним з таких є package.json. Головна задача цього файлу це збереження інформації про розробника, версію програмного забезпечення, певний загальний опис функціоналу та набір модулів та плагінів, які використовуються для розробки та запуску проекту. Наявність цього файлу в проекті слугує документацією про сторонні бібліотеки та модулі. Також, розробнику не потрібно власними руками шукати та встановлювати плагіни. Якщо вони є у цьому файлі, то у розробника є можливість їх швидко

завантажити на своє локальне середовище за допомогою команди в консолі «npm install».

В цьому файлі усі залежності поділені на dev залежності та звичайні. До dev залежностей зазвичай належать плагіни та модулі, які використовуються під час розробки або тестування; такі залежності, що не використовуються власне при деплої проекту на сервері (рис 3.6). До таких плагінів належать: babel та його допоміжні модулі, css-loader, file-loader, style-loader, url-loader, json-loader, html-webpack-plugin. Babel конвертує ES6+ в більш стару версію, яка зараз використовується в сучасних браузерах та за допомогою цього модулю застосовуються усі сучасні методи та особливості мови JavaScript. Модулі «*-loader» – це спеціальні конфігурації для бандлеру webpack. За допомогою цих лоадерів можна кастомізувати бандлер та задати правила для обробки файлів, їх мінімізацію, перевірку та інші можливі інструкції перед створення результуючих файлів. Webpack – це інструмент для збору проекту та зручної локальної розробки за рахунок власного локального серверу.

Рисунок 3.6. Набір dev залежностей проекту в файлі package.json

До звичайних залежностей належать модулі та плагіни, які використовуються в самому проекті та без яких проект буде неправильно працювати або некоректно видавати певну інформацію (рис. 3.7.). В нашому проекті такими плагінами є: react, react-autocomplete, material-ui, react-bootstrap, react-dom, react-input-mask та react-select. React – це основна бібліотека в якій увесь основний функціонал. React-dom – допоміжний файл для маніпуляції з dom веб-сторінки. React-autocomplete, react-select та react-input-mask – бібліотеки для створення зручного інтерфейсу форм, селект-елементів та валідації полів введення інформації. React-bootstrap – це аналог css фреймворку bootstrap з додатковою реалізацією елементів у вигляді компонентів.

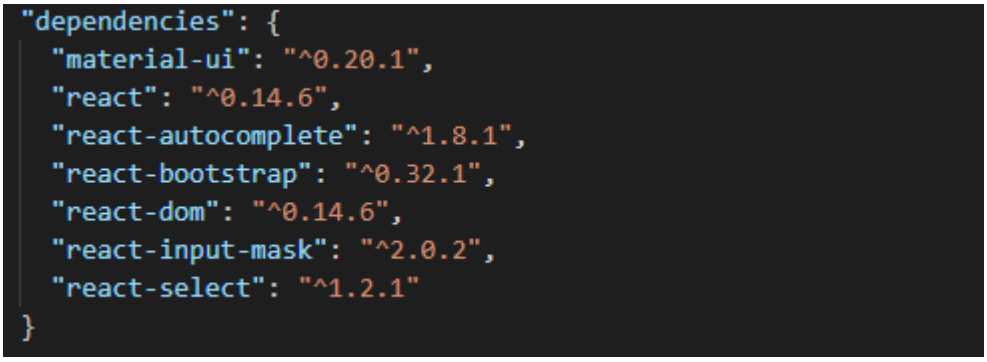


Рисунок 3.7. Набір залежностей проекту в файлі package.json

Також для швидкого виконання команд, які виконуються під час розробки проекту велику кількість разів були створені npm скрипти. Команди, які часто використовуються в нашому проекті відносяться до розгортання проекту в локальному середовищі на сервері, створення так званих build-файлів в різних модифікаціях (режим production мінімізує файли).

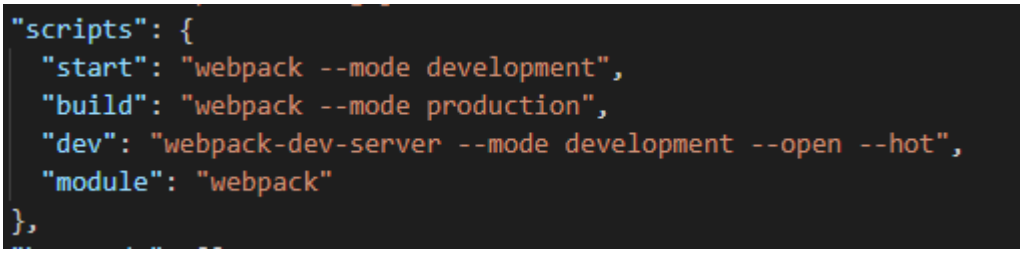


Рисунок 3.8. Набір npm скриптів системи

3.4. Розробка REST API

Після створення структури бази даних та заповнення таблиць даними був створений механізм доступу до цих даних. В нашому випадку було створено Rest API. Суть API в доступі до бази даних за певними ендпоінтами, в яких закладена логіка доступу до необхідних даних. Також в API було закладено розпізнавання типу користувача та обмеження доступу до даних у випадку, якщо права користувача були перебільшені (наприклад, обмеження в заповненні таблиці рейтингу для студентів). Дана можливість реалізована за допомогою middleware. Middleware – це спеціальний функціонал для API, який виконується прямо перед виконанням та відправкою запиту. Часто middleware використовується для перевірки прав доступу за певним ендпоінтом, мемоізації або іншими словами механізму кешування даних та пришвидшення швидкодії роботи додатку. Перед відправкою запиту проводиться перевірка прав користувача. Якщо доступ до цього URL в межах прав користувача, то запит буде відправлений. В іншому випадку, коли користувачу не вистрачає прав для доступу за певним URL, то користувач отримає відповідне повідомлення про неможливість доступу до цієї сторінки.

```php
Route::group(['middleware' => 'guest'], function () {

    Route::get('/', 'Auth\LoginController@showLoginForm');

    Route::post('/login', 'Auth\LoginController@login');
    Route::get('/confirm/{code}','ResetController@confirmEmail')->name('confirm');
});

Route::group(['middleware' => 'auth', 'prefix' => 'personal-office'], function () {

    Route::get('/', 'MainController@index')->name('home');

    Route::group(['prefix'=>'teasers'],function(){
        Route::get('/', 'TeaserController@getTeasers');
        Route::post('/edit', 'TeaserController@edit')->middleware('admin');
        Route::post('/create', 'TeaserController@create')->middleware('admin');
        Route::post('/destroy', 'TeaserController@destroy')->middleware('admin');
    });

    Route::group(['prefix' => 'profile'],function (){
        Route::get('/', 'UserController@showDataAboutUser')->name('profile');
        Route::get('/get-data','UserController@getDataForPersonalOffice');
        Route::post('/edit-data','UserController@updatePersonalOffice');
        Route::post('/save-avatar','UserController@saveUserAvatar')->name('avatar');
        Route::post('/change-password','UserController@changeUserPassword');
        Route::post('/reset-user-password/{user_id}','UserController@resetUserPasswordByAdmin')->middleware('admin');
    });
```

Рисунок 3.9. Приклад ендпоінтів REST API для рейтингової системи

На рисунку можна побачити, що middleware створює групи ендпоінтів та додає для них новий окремий від задачі ендпоінтів функціонал. Наприклад, для усіх ендпоінтів, які необхідні користувачу для зручного використання та налаштування особистого кабінету створений middleware функцію auth, яка визначає до якої категорії користувачів належить студент та відповідно надає доступ до функціоналу. Також в middleware передається параметр prefix, який визначає та додає строку з параметру до початку URL. Таким чином, при параметрі префіксу «personal-office» ми маємо наступний кінцевий URL: personal-office/edit, personal-office/create і т.д.

Пригадаємо усі існуючі на даний момент HTTP методи, які використовуються при розробці клієнт-серверного додатку (рис. 3.10).

HTTP методи	Опис
Head	Відправка header заголовків запитів
Trace	Пересилання запитів, які були отримані
Put	Оновлення файлів на сервері
Delete	Видалення файлів з серверу
Options	Визначає які HTTP методи підтримуються сервером та відправляє відповідну помилку
Post	Додавання файлів на сервер
Get	Отримання даних з серверу

Рисунок 3.10. Список існуючих HTTP методів запитів

Взагалі існує 7 HTTP методів запитів: Head, Trace, Put, Delete, Options, Post, Get. Head дозволяє передавати header інформацію о запиті, тобто набір заголовків, які супроводжують обмін даними з сервером. До таких заголовків належать заголовки про кодування, мову, cookies та кешування. Trace використовується для пересилання запитів, які були отримані. Put використовується для оновлення даних, які вже є на сервері або створення даних на сервері, якщо унікальний ідентифікатор не був знайдений. Delete –

це метод який може видалити файл або певні дані з таблиці бази даних. Options несе службову інформацію та вказує які HTTP методи запитів підтримуються сервером та вертає у якості відповіді відповідну помилку. Post метод створений для генерації нових даних на сервері. Get створений для отримання необхідних даних з серверу. Найчастіше в нашій рейтинговій системі використовувалися Get та Post методи запитів.

Також можна побачити які методи запитів були створені. Route::get створює HTTP запит на доступ к даним. Наприклад, get метод за ендпоінтом /get-data створює запит до бази даних та при успішному статусі запиту завантажує дані про користувача. При користуванні get запитом дані в базі залишаються незмінними. За кожним з ендпоінтів затверджений свій контролер, який зберігає в собі набір функцій об'єднаних за певною тематикою та ціллю.

Наступним кроком для оптимізації запитів правильним рішенням було створити систему кешування даних. Наша система виконана у вигляді middleware – певний функціонал, який виконується перед відправленням запиту. Працює система наступним чином, користувач відправляє запит за певним ендпоінтом. Далі перевіряється роль користувача та можливість відправити користувачу цей запит. Після успішної перевірки система кешування перевіряє чи відправляли такий запит раніше.

```
Route::group(['prefix'=>'teasers'],function(){
    Route::get('/', 'TeaserController@getTeasers');
    Route::post('/edit', 'TeaserController@edit')->middleware('admin');
    Route::post('/create', 'TeaserController@create')->middleware('admin');
    Route::post('/destroy', 'TeaserController@destroy')->middleware('admin');
});
```

Рисунок 3.11. Список ендпоінтів та контролерів модулю новин

У випадку коли запит раніше не відправлявся наступним кроком відбувається запит до бази даних та отримуються необхідні дані. Після цього створюється локальний об'єкт, де кожен рядок представляє запит, де ключ це

ендпоінт, а значенням є результати запиту. При наступному запиті перевіряється локальний об'єкт і при співпадінні ключа з об'єкту з ендпоінтом, запит до бази не відправляється, а необхідні значення отримуються з локального об'єкту.

Розглянемо модуль новин та оголошень. Можемо побачити, що було створено чотири ендпоінти: один get запит для відображення усіх новин та оголошень та три post запити на зміну створення та видалення даних на серверній частині. Також можна помітити додаткову перевірку за допомогою middleware admin для ендпоінтів /edit, /create та /destroy, яка перевіряє користувача на наявність прав адміністратора. Для цієї групи ендпоінтів був створений контролер TeaserController. Через спеціальний символ @ можна побачити який саме метод контролеру виконується при виконанні запиту.

```php
public function getTeasers()
{
    $greetings = $this->getTeaserBuilder(1,'greetings');
    $promo = $this->getTeaserBuilder(1,'promo');
    $articles = $this->getTeaserBuilder(2,'article');
    return response([
        'greetings' => $greetings,
        'promo' => $promo,
        'articles' => $articles,
        'can_edit' => Auth::user()->type === 'admin',
    ]);
}

public function edit(Request $request)
{
    $teaser = Teaser::findOrFail($request->id);
    $data = $request->all();
    unset($data['image']);
    if($request->hasFile('image')) {
        $data['image'] = parent::saveImage($request->file('image'), $teaser->image,
            Teaser::DEFAULT_IMAGE, 'teasers');
    }
    return response($teaser->map($data));
}

public function create(TeasersRequest $request){
    $teaser = new Teaser();
    $data = $request->all();
    if($request->hasFile('image'))
        $data['image'] = parent::saveImage($request->file('image'),Teaser::DEFAULT_IMAGE,
            Teaser::DEFAULT_IMAGE,'teasers');
    return response($teaser->map($data));
}

public function destroy(Request $request)
{
    Teaser::findOrFail($request->id)->delete();
}
```

Рисунок 3.12. Контролер TeaserController з модулю новин та оголошень

Детальніше розглянемо контролер TeaserController, який використовується в модулі новин та оголошень (рис.3.12). Контролер TeaserController є класом, який наслідується від класу Controller. Представляє собою набір функцій для маніпуляції з даними в модулі новин та оголошень. В даному контролері створені наступні функції: getTeasers для відображення усіх новин, метод edit, який створений для редагування існуючих новин, метод create для створення нової публікації та destroy для знищення існуючої новини. Також можна помітити допоміжний метод findOrFail в методах edit та destroy, який перевіряє на наявність вхідний параметр та повертає перший результат виклику або у випадку відсутності інформації повертається відповідний exception.

3.5. Створення автоматизованих тестів рейтингової системи

Зі зростанням функціоналу та можливостей системи зростає й кількість коду проекту. На початковому етапі можна зупинитися на так званому мануальному тестуванні. Для того, щоб автоматизувати цей процес та протестувати функціонал проекту створимо unit тести. Суть цих тестів полягає в наступному: беремо методи компонентів або класів, прогнозуючи які значення можуть бути в якості вхідних даних та що ми отримаємо на виході, описуємо відповідну функцію. Після створення тестів за допомогою консольних команд ці тести виконуються.

В результаті виконання цих тестів ми маємо повідомлення, яке повідомляє чи співпадають результати, які прогнозувались та були очікувані та актуальні вихідні дані. Достатньо детально описаний тест може замінити документацію до проекту. Для розробки тестів для нашої рейтингової системи була застосована бібліотека PHPUnit.

Одним з прикладів використання unit-тестів є перевірка функцій, які пов'язані з алгоритмом підрахунку загального рейтингового балу студентів. Для даного функціоналу був створений окремий клас UserRaitingTest та

створені чотири методи для детального тестування усіх можливих ситуацій під час розрахунку рейтингу користувача. Усі чотири методи перевіряють дані студентів з різною кількістю балів. Для кожної функції, яка буде слугувати unit-тестом необхідно викликати метод assertEquals. Ця функція порівнює значення, які були передані у вигляді параметрів. Першим параметром є значення яке повинно бути, другим параметром передається фактичне значення, яке створює зараз функціонал системи. Якщо тест успішно виконався, то можна зробити висновок, що наш функціонал працює за нашим сценарієм, який ми описали та який ми очікуємо.

Після запуску наших unit-тестів маємо такий результат (рис. 3.13).

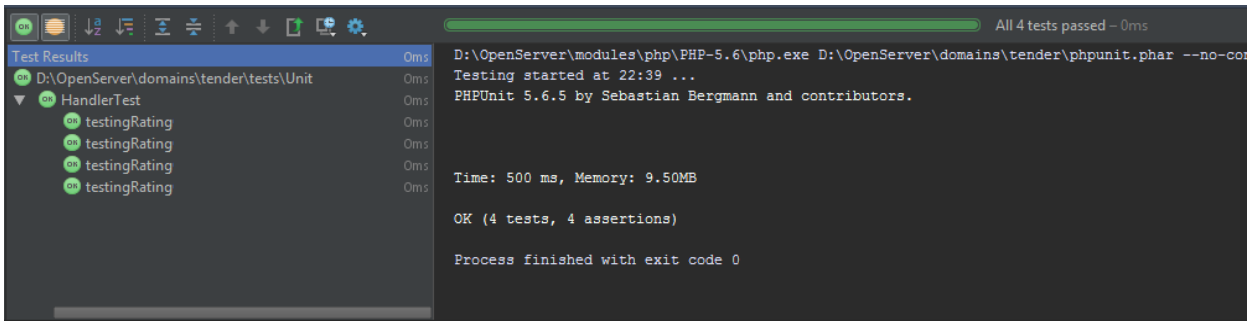


Рисунок 3.13 – Приклад успішного виконання unit-тестів

Як ми можемо побачити на рисунку усі unit-тести, які виконують перевірку формування рейтингової системи були виконані та пройдені успішно. Також на рисунку ми можемо побачити додаткову інформацію про проходження тестів. В цій консолі ми можемо побачити, що усі тести виконались зі статусом OK. Також є додаткова інформація про час, за який виконалися тести та пам'ять, яка була виділена на ці тести. Великі показники цих даних можуть показувати на зациклення в коді або витік пам’яті, які можуть призвести до збоїв в роботі системи. Для демонстрації помилкової ситуації створимо умову в одному з тестів, яка за будь-яких умов не виконається (рис.3.14). Можемо побачити, що помилку у виконання тесту можна побачити як у лівому меню, так і у шкалі справа. Також можемо побачити повідомлення в консолі, яке демонструє актуальний результат та результат, який очікувався.

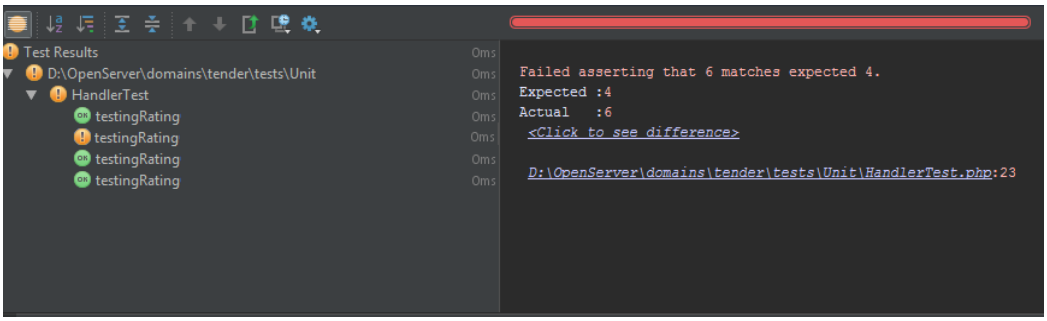


Рисунок 3.14. Приклад проходження тестів з помилкою

3.6. Аналіз результатів роботи та швидкодії системи

Проаналізуємо швидкодію нашої рейтингової системи навчання студентів з дисципліни робототехніка. Скористаємося консоллю розробника браузеру Google Chrome та розглянемо показники завантаження сторінок системи.

Завантаження головної сторінки, яка в нашому випадку являється сторінкою з новинами та оголошеннями займає близько двох секунд (рис. 3.15).

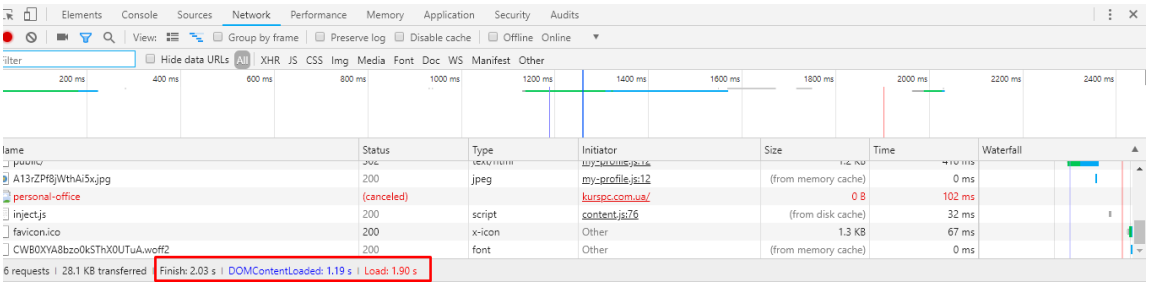


Рисунок 3.15. Показники завантаження головної сторінки рейтингової системи

Завантаження сторінки з завданнями для користувачів теж займає близько двох секунд.

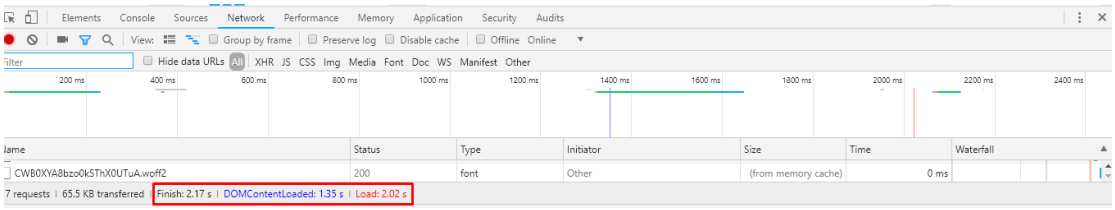


Рисунок 3.16. Показники завантаження сторінки рейтингової системи з завданнями для студентів

Також проаналізуємо показники завантаження сторінки, на якій відображені результатів студентів у вигляді таблиці успішності (рис. 3.17).

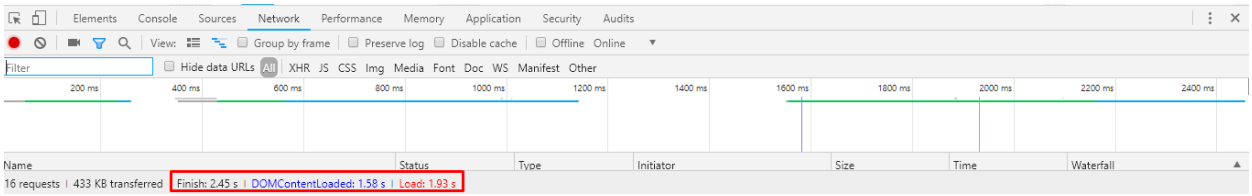


Рисунок 3.17. Показники завантаження сторінки з успішністю студентів

Отже, проаналізувавши всі отримані результати, можна зробити висновок, що система має належну швидкодію та завдяки цьому є дуже зручною у користуванні.

Висновки до розділу

В цьому розділі був проведений опис архітектури рейтингової системи навчання студентів. Система була розділена на 5 модулів. Весь процес роботи та функціонал модулів були детально описані. Для забезпечення зручної та швидкої роботи з даними користувачів системи була описана структура та створена база даних. Проведена розробка REST API для розділення запитів за відповідними адресами за призначенням користувача. Був проведений детальний аналіз технологій та методів, які можна використати для оптимізації на даному етапі. Запити до баз даних, які створені з метою отримання інформації про завдання або набір інформації про усіх користувачів для створення рейтингової системи поділені на частини. Таким чином інформація буде швидше завантажуватись та відображатись у браузері. Був створений на серверній стороні middleware, який відповідає за мемоїзацію даних. Таким чином була зменшена швидкість завантаження сторінки для даних, які вже завантажувались раніше. Для тестування системи були залучені методики мануального та автоматичного тестування. Зі сторони автоматичного тестування були створені unit-тести, які мають можливість неодноразово

перевірити функціонал системи та зробити невеликий звіт, який вказує на статус після виконання тестів, час та на помилки, якщо вони присутні. Зі сторони мануального тестування була перевірена клієнтська частина інформаційної системи. Система була перевірена на адаптивність на популярних роздільних здатностях.

РОЗДІЛ 4. МАРКЕТИНГОВИЙ АНАЛІЗ СТАРТАП-ПРОЕКТУ

В цьому розділі був проведений маркетинговий аналіз проекту та представлення його у вигляді стартапу. Опишемо можливості впровадження системи на ринку.

4.1. Опис ідеї проекту

Основна ідея стартап проекту – це створення рейтингової системи, яка буде відрізнятися своєю простотою в користуванні, надійністю та швидкодією. Рейтингова система демонструє прогрес людей у навчанні з певної сфери знань. Важливим фактором даної системи є зручність у користуванні та належна її швидкодія. Ці особливості повинні створити конкурентну здатність на рейтингу інформаційних систем.

Опишемо детальніше саму ідею, застосування та вигоди користувачів, які вони можуть отримати (табл. 4.1).

Таблиця 4.1 Опис ідеї стартап проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Розробити рейтингову систему оцінювання студентів як підсистему для навчання студентів	Використання навчальними закладами та компаніями, які займаються навчанням людей для ведення результатів студентів, груп, викладачів.	Приватні навчальні компанії та державні навчальні заклади використовують зручний у використанні продукт, який надає можливість спростити та автоматизувати роботу
	Застосування користувачами в навчальних цілях студентами для навчання за рахунок перегляду відеоматеріалів	Студенти отримують можливість навчатися онлайн, мають постійний доступ до навчальних матеріалів та можуть переглядати свої результати навчання з того чи іншого предмету
	Використання работодавцями для моніторингу студентів у певних спеціалізованих напрямках навчання	При відповідній популярності системи можна швидко проводити відбір у компанію за допомогою проходження тестових завдань

Наступним кроком оцінки ідеї є опис техніко-економічних переваг, які має даний додаток в порівнянні із вже існуючими рішеннями. В якості найближчих аналогів нашої інформаційної системи були обрані системи ITVDN та Web Academy. Визначимо сильні, слабкі та нейтральні характеристики нашого додатку в порівнянні з обраними конкурентами (табл. 4.2).

Таблиця 4.2 Визначення сильних, слабких та нейтральних характеристик ідеї проекту

№	Техніко-економічні характеристики ідеї	Продукція конкурентів			W (слабка сторона)	N (нейтральна сторона)	S (сильна сторона)
		Мій проект	ITVDN	Web Academy			
1	Сучасний дизайн	+	+	+		+	
2	Зручність у користуванні	+	-	+		+	
3	Швидкодія роботи	+	-	-			Має достатньо високу та конкурентноздатну швидкодію роботи.
4	Можливість використання методичних матеріалів	+	-	-		+	+
5	Формування груп та уроків в системі	+	-	-			Створює можливість вчителю та адміністратору формувати завдання та таблиці оцінок

6	Виставлення оцінок та формування рейтингу групи	+	-	-			Усі студенти отримують оцінки за уроки. Також вони мають змогу моніторити усі свої результати
7	Відсутність нагромадження інформації на сторінці	+	-	-			+
8	Масштабування системи	+	+	+		+	

В цій таблиці були знайдені та описані слабкі, сильні та нейтральні сторони нашої інформаційної системи. За цими даними ми можемо зробити висновок про конкурентоспроможність системи, що даний проект має переваги над своїми аналогами.

4.2. Технологічний аудит ідеї проекту

Зробимо перегляд технологій, які можна використати в розробці ідеї проекту (табл. 4.3).

Таблиця 4.3 Технології здійснення ідеї проекту

№	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
1	Розробка рейтингової системи для навчання студентів та керування навчальним процесом і забезпечення її належної швидкодії	Застосування мови програмування PHP	Наявна. Має добре продумані фреймворки.	Вільна
		Застосування мови програмування JavaScript	Наявна. Має велику кількість готових рішень для вирішення різних проблем. Має декілька фреймворків для роботи з веб-додатками	Вільна
		Застосування мови програмування C#	Наявна. Має один фреймворк для роботи з веб-додатками	Вільна

		Застосування мови програмування Python	Наявна. Має один фреймворк для роботи з веб-додатками	Вільна
Обрана технологія реалізації проекту: JS(React).				

Переглянувши сучасні технології та зробивши висновок про технології, які будуть використовуватись для створення рейтингової системи можна сказати, що для реалізації клієнтської частини найкраще підходить мова програмування JavaScript, оскільки вона найбільше підходить для релізації ідеї односторінкового веб-додатку з належною швидкодією. Дана мова та фреймворк React вже багато років існує на ринку мов програмування, тому має велику кількість готових рішень в області розробки веб-додатків.

4.3. Аналіз ринкових можливостей запуску стартап-проекту

Опишемо ринкові можливості, які можуть використовуватись та ринкові загрози, які можуть заважати становленню проекту.

Таблиця 4.4. Попередня характеристика потенційного ринку стартап проекту

№	Показники стану ринку	Характеристика
1	Кількість головних гравців, од	10
2	Загальний обсяг продаж, грн./ум.од	2000 в рік
3	Динаміка ринку	Зростає
4	Наявність обмежень	Відсутні
5	Специфічні вимоги до стандартизації та сертифікації	Відсутні
6	Середня норма рентабельності в галузі або по ринку, %	100%

Орієнтуючись на дані цієї таблиці можна прийти до висновку, що ринок реалізації товару є привабливим та присутня велика ймовірність реалізації цієї ідеї.

Визначимо потенційних клієнтів даного стартап проекту (табл. 4.5.).

Таблиця 4.5. Характеристика потенційних клієнтів стартап проекту

№	Потреба, що формує ринок	Цільова аудиторів(цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1	Наявність рейтингової системи для навчання студентів, засіб для автоматизації процесів інформаційної системи	Цільовою аудиторією є веб-студії та навчальні заклади, які проводять навчання студентів за різними напрямами та бажають автоматизувати процес та завжди мати під рукою всі потрібні дані про студентів та навчальний процес та можливі роботодавці, які мають певний специфічний процес відбору до своєї компанії такі як певна кількість тестових завдань.	Веб-студії та навчальні заклади в основному однаково зацікавлені у використанні даного веб-додатку, оскільки він дозволяє автоматизувати їх роботу	Користувач повинен бути ознайомлений з дизайнами сучасних інформаційних систем та мати постійний доступ до інтернету для користування даним продуктом

Описавши характеристики потенційних клієнтів, можна сказати, що головними користувачами даної системи можуть стати приватні компанії, які надають навчальні послуги або державні навчальні заклади з потребами в спрощенні робочого процесу. Потенційними клієнтами можуть стати великі компанії-роботодавці, які зацікавлені в наймі студентів з гарною навчальною базою, але для цього систему необхідно доопрацювати.

Наступним етапом буде проведення аналізу ринкового середовища. До аналізу належать фактори, які представляють загрозу та потенційно можуть вплинути на реалізацію продукту на ринку та його продажі (табл. 4.6).

Таблиця 4.6 Фактори загроз

№	Фактор	Зміст загрози	Можлива реакція компанії
1	Погіршення швидкодії роботи веб-додатку при великій кількісті даних	Робота системи відповідає нормам швидкодії, які були визначені під час розробки проекту. Зі збільшенням кількості даних в таблиці є загроза погіршення швидкодії роботи та як наслідок сповільнення системи	Звернення до людей, які відповідальні за розробку продукту та її обслуговування для вирішення проблеми
2	Вірогідність появи певних помилок в системі	Дана система має зв'язок між певними її компонентами. Тому для коректної роботи системи необхідно забезпечити систему відповідними вхідними даними.	Більш детальний опис документації, створення відео уроків, які демонструють як правильно користуватися системою

Проведемо огляд факторів можливостей, які можуть очікувати додаток при виході на ринок (табл. 4.7).

Таблиця 4.7 Фактори можливостей

№	Фактор	Зміст можливості	Можлива реакція компанії
1	Автоматизація робочого процесу	Дана система дозволяє оптимізувати роботу навчального закладу за рахунок зменшення часу на створення груп студентів, завдань та таблиць успішності студентів.	Створення спеціальних інструкцій для користувачів з метою підвищення розуміння системи
2	Монетизація системи	Система дозволяє отримувати певні кошти за допомогою залучення компаній, які потребують проведення спеціалізованого відбору студентів	Надання відповідного функціоналу та доступів для моніторингу результатів рейтингової системи

Далі проведемо ступеневий аналіз конкуренції(табл. 4.8).

Таблиця 4.8 Ступеневий аналіз конкуренції на ринку

№	Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства(можливі дії компанії, щоб бути конкурентноспроможною)
1	Тип конкуренції: чиста конкуренція	Існує велика кількість конкурентів, які пропонують розробку системи даного типу	Реклама, підвищення швидкодії, якості та зручності у користуванні системою
2	Рівень конкурентної боротьби: національний	Рішення може використовуватися в навчальних закладах та веб-студіях на території України	Розширення функціональних можливостей системи та її доопрацювання.
3	Галузева ознака: внутрішньогалузевий	Конкуренція в сфері навчання студентів та автоматизації навчального процесу	Покращення якості та зручності в роботі системи та її функціональне розширення
4	Конкуренція за видами товарів: товарно-видова	Дана конкуренція - конкуренція між товарами одного виду	Впровадження функціональності, яка відсутня у додатках інших розробників. Спрощення в роботі додатку та забезпечення його належної якості.
5	Характер конкурентних переваг: цінова	Співвідношення вартості системи до її надійності та якості	Продаж системи за адекватні кошти та забезпечення належної підтримки
6	За інтенсивністю: марочна	Наявність унікального зразка, який відрізняє даних продукт від продуктів-замінників	Впровадження власної назви та власного знаку.

Таблиця 4.9 Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товарозамінники
	Веб студії, які мають велику популярність та ініціативи навчальних закладів	Інші дистриб'ютори	Відсутні	Приватні та державні навчальні заклади	Відсутні
Висновки	Необхідно надавати унікальний фунціонал, забезпечити належну швидкодію та зручність у користуванні системою	При розширенні та доопрацюванні функціональності системи може стати конкурентом.	Відсутні	Кожний клієнт має певні побажання, які доцільно враховувати при розробці системи	Відсутні

Аналізуючи минулі дослідження, було визначено, що на ринку у даного продукту присутня конкуренція. Кожна з систем виконана на непоганому рівні, але кожна з них має певні помилки. Також продукт стартапу має функціонал, який не був помічений у конкурентів. При уникненні помилок конкурентів та створенні зручної та швидкої системи можна досягнути значної кількості продажів.

Таблиця 4.10 Обгрунтування факторів конкурентоспроможності

№	Фактор конкурентоспроможності	Обгрунтування
1	Автоматизація навчального процесу	За рахунок автоматизації навчального процесу відбувається зменшення часу необхідного для ведення обліку груп, студентів та інших даних певного навчального закладу
2	Належна швидкодія системи	Система працює швидше за своїх конкурентів та забезпечує швидке відображення потрібної користувачу інформації
3	Масштабування системи	Система може бути легко масштабована та витримувати значне навантаження та велику

		кількість користувачів, які одночасно нею користуються

Таблиця 4.11 Порівняльний аналіз сильних та слабких сторін системи

№	Фактор конкуренто-спроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні з новою системою						
			-3	-2	-1	0	+1	+2	+3
1	Автоматизація навчального процесу	10			+				
2	Належна швидкодія системи	15		+					
3	Масштабування системи	10			+				

На основі отриманих даних, сформуємо SWOT – аналіз (матриці аналізу сильних Strength та слабких Weak сторін, загроз Troubles та можливостей Opportunities на основі виділених ринкових загроз та можливостей, а також сильних та слабких сторін). Перелік ринкових загроз та можливостей складається на основі аналізу факторів загроз та факторів можливостей маркетингового середовища. Ринкові загрози та можливості є наслідками впливу факторів. Цей метод можна вважати універсальним, бо він використовується для аналізу будь-яких об’єктів дослідження. Недоліком цього методу можна вважати той факт, що SWOT демонструє лише перелік певних факторів, не виявляючи в них основні та другорядні фактори або взаємного зв’язку.

Таблиця 4.12. SWOT-аналіз стартап-проекту

Сильні сторони (S): • належна швидкодія • зручність у користуванні • масштабованість	Слабкі сторони(W): • вартість розробки • вартість обслуговування
Можливості(O): • розширення функціоналу	Загрози(T): • недостатнє фінансування

• покращення роботи системи	• поява конкурентів • відсутність компаній, які будуть зацікавлені в користуванні веб-додатку

На основі SWOT-аналізу розробимо альтернативи ринкової поведінки для забезпечення виведення стартап проекту на ринок та орієнтовний оптимальний час їх ринкової реалізації з огляду на потенційні проекти конкурентів, що можуть бути також виведені на ринок. Визначення альтернатив здійснюється з точки зору строків та ймовірності отримання оптимальних ресурсів.

Таблиця 4.13 Альтернативи ринкового впровадження стартап-проекту

№	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Безкоштовне надання певного функціоналу у користування споживачам на обмежений термін	Головний ресурс – люди, даний ресурс - наявний	1-3 місяці
2	Реклама	Залучення власних коштів для реклами товару	2-3 місяці
3	Написання статей та опис товару на відомих ресурсах	Головний ресурс – час, даний ресурс - наявний	2-3 тижні
4	Презентація товару на конференціях та інших IT заходах	Ресурс – час та гроші для участі, наявні	1 місяць

В результаті проведено аналізу, можна зробити висновок, що найбільш привабливою альтернативою для ринкового впровадження стартап проекту, є презентація товару на конференціях та інших IT заходах.

4.4. Розроблення ринкової стратегії проекту

Розроблення ринкової стратегії першим кроком передбачає визначення стратегії охоплення ринку. Для цього необхідно визначитися з цільовою аудиторією системи, які є конкуренти на даному ринку та як швидко можна увійти на даний ринок, актуальність цього продукту (готовність споживачів прийняти продукт).

Таблиця 4.14. Вибір цільових груп потенційних споживачів

№	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Програмісти. Вік: від 18 до 40. Місце проживання: не важливо. Сімейний стан: не важливий. Сфера зайнятості та рівень заробітної плати: IT сфера, від 30 тис. грн.	Даний продукт можна використовувати як засіб автоматизації роботи навчального закладу	1 ліцензія для навчального закладу чи веб-студії	Інтенсивність конкуренції в сегменті висока, оскільки існує великий попит систем даного типу	Сегмент дозволяє вийти на ринок та показати переваги даного продукту у контексті продуктів-аналогів

2	Підприємства. Сфера зайнятості – навчання студентів.	Система дозволяє автоматизувати роботу навчального закладу або веб-студії та забезпечити хороший механізм для навчання студентів.	1 ліцензія для навчального закладу або веб-студії	Інтенсивність конкуренції в сегменті висока, оскільки існує велика кількість виробників схожих продуктів.	Сегмент дозволяє вийти на ринок та показати переваги даного продукту у контексті продуктів-аналогів
Які цільові групи обрано: ІТ сфера та навчальні заклади, у яких виникає необхідність в автоматизації навчального процесу та керуванням ним.					

Оскільки цільовою групою виступають навчальні заклади та розробники різних сфер, оберемо стратегію масового маркетингу.

Таблиця 4.15 Визначення базової стратегії розвитку

Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
Надання функціональності що відсутня у товарів-замінників, підтримка клієнтів	Проведення реклами, освітлення унікальної функціональності через інтернет ресурси та інші канали, контакт напряму з споживачами; формування лояльності і прихильності споживачів	Зниження ступеню замінності товару; Прихильність клієнтів; Відмітні властивості товару; Відмітні характеристики товару;	Стратегія диференціації

Таблиця 4.16 Визначення базової стратегії конкурентної поведінки

Чи є проект «першопрохідцем» на ринку	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, які?	Стратегія конкурентної поведінки

Ні, оскільки є товари-аналоги, але дані товари не мають достатньої функціональності	Так, ціль компанії знайти нових споживачів та, частково, переманити існуючих у конкурентів задля задоволення потреб останніх	Компанія частково копіює характеристики товару конкурента, основна ціль компанії розробка нового унікального функціоналу, з підтримкою основного функціоналу конкурентів	Стратегія заняття конкурентної ніші

Таблиця 4.17 Визначення стратегії позиціонування

№	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту
1	Відмінні властивості товару	Стратегія диференціації	Унікальність функціоналу;	Надійність Актуальність Швидкодія
2	Підтримка з боку розробника	Стратегія диференціації	Підтримка клієнту; Ліцензії; Модернізація системи	Клієнтоорієнтованість
3	Відповідність загальновживаним інтерфейсам	Стратегія диференціації	Підтримка та вдосконалення сучасних всім знайомих методів	Стабільність Зручність

Тобто, у якості базової стратегії розвитку було обрано стратегію диференціації та стратегію заняття конкурентної ніші, яка має базову стратегію конкурентної поведінки.

4.5. Розроблення маркетингової програми стартап-проекту

Таблиця 4.18 Визначення ключових переваг концепції потенційного товару

№	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити)
1	Автоматизація роботи веб-студії або навчального закладу	Підвищення ефективності роботи веб-студії або навчального закладу за рахунок автоматизації навчального процесу	Існуючі конкуренти мають меншу швидкодію системи та обмежений функціонал
2	Можливість моніторингу результатів навчання студентів	Студенти мають змогу контролювати свої результати та бачити результати виконання контрольних завдань	Даних систем рейтингового оцінювання на ринку не спостерігається

Здійснимо опис рівнів моделі товару.

Таблиця 4.19 Опис трьох рівнів моделі товару

<table>
<tr><td>Рівні товару</td><td colspan="3">Сутність та складові</td></tr>
<tr><td>1. Товар за задумом</td><td colspan="3">Забезпечення автоматизації роботи рейтингової системи веб-студії чи навчального закладу та надання роботодавцям долучатися до системи та проводити відбір до своєї компанії</td></tr>
<tr><td rowspan="6">2. Товар у реальному виконанні</td><td>Властивості/характеристики</td><td>М/Нм</td><td>Вр/Тх/Тл/Е/Ор</td></tr>
<tr><td>Зручність використання</td><td>-</td><td>Висока</td></tr>
<tr><td>Забезпечення швидкої роботи та обробки даних</td><td>Час виконання запитів</td><td></td></tr>
<tr><td>Вартість</td><td>Грн.</td><td>0</td></tr>
<tr><td colspan="3">Якість: відповідність загальновживаним нормам</td></tr>
<tr><td colspan="3">Пакування: ліцензія на використання системи</td></tr>
<tr><td colspan="3">Марка: myRating</td></tr>
<tr><td>3. Товар із підкріпленням</td><td colspan="3">До продажу: наявна повна документація, акції на придбання декількох ліцензій</td></tr>
</table>

	Після продажу: підтримка та доопрацювання продукту в разі необхідності та бажання замовника
Проект буде захищено від копіювання за рахунок реєстрації назви програми, отримання патенту на програмний код даного продукту та окремих алгоритмів обробки даних	

Таблиця 4.20 Визначення меж встановлення ціни

Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
інформація відсутня	приблизно 60 тис. грн..	від 50 тис. грн.	30-50 тис. грн

Таблиця 4.21 Формування системи збуту

Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
Отримання продукту	Ліцензії	Нульовий та/або однорівневий	Традиційна

Таблиця 4.22 Концепція маркетингових комунікацій

№	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуютьс я цільові клієнти	Ключові позиції, обрані для позиціонува ння	Завдання рекламного повідомлення	Концепці я рекламно го звернення
1	Отримання продукту / функціональності	Електронна пошта, телефон, факс, електронна форма на сайті	Унікальні властивості товару	Привернути увагу клієнтів, освітити унікальну функціональність	Творча та класична
2	Отримання підтримки від компанії		Якісна підтримка	Дати зрозуміти що клієнт може розраховувати на підтримку зі сторони розробника	

В результаті було створено ринкову програму, що включає в себе визначення ключових переваг концепції потенційного товару, опис моделі

товару, визначення меж встановлення ціни, формування системи збуту та концепцію маркетингових комунікацій.

Висновки до розділу

В даному розділі було описано ідею проекту, здійснено аналіз потенційних техніко економічних переваг ідеї, а саме визначено сильні, нейтральні та слабкі сторони даної ідеї. Проведено аудит описаної ідеї та визначення її технологічної здійсненності. На основі отриманих результатів було зроблено висновок, що дана ідея є здійсненною. Виконано попередню оцінку потенційного ринку стартап проекту, визначено потенційних клієнтів даного продукту, визначено основні фактори загроз та можливостей. Проведено ступеневий аналіз конкуренції на ринку, обгрунтовано фактори конкурентоспроможності та виконано порівняльний аналіз сильних та слабких сторін продукту та конкурентів. Проведено SWOT-аналіз та альтернативні шляхи впровадження даного продукту. На основі отриманих результатів можна зробити висновок, що даний проект може бути впроваджений та зайняти свою нішу.

ВИСНОВКИ

В ході роботи над магістерською дисертацією були визначені мета роботи, предмет та об’єкт дослідження. Був проведений аналіз інформаційних систем, які вже існують на ринку та які можна вважати аналогом за підходом або функціоналом. Було прийнято рішення орієнтуватися на наступні дві інформаційні системи: Web Academy та ITVDN. Ці системи найбільш схожі за функціоналом з системою, яку нам необхідно було створити. Були визначені переваги та недоліки цих систем, взяті показники швидкодії.

Був проведений опис технологій та їх вибір для розробки проекту. Для клієнтської частини був задіяний фреймворк React. В статті був проведений аналіз існуючих фреймворків та після оцінки усіх переваг та недоліків було прийнято рішення використати саме фреймворк React.

В наступному розділі була визначена архітектура проекту та описані основні модулі системи. Створена структура бази даних, яка налічує 9 таблиць. Були створені таблиці користувачів, завдань, груп користувачів, новин та оголошень для користувачів та інші таблиці, які необхідні для створення зв’язків та збереження додаткової інформації. Створено REST API, яке зв’язує клієнтську та серверну частину. Були створені ендпоінти, які повністю покривають усі запити до серверу. Була розроблена система кешування, яка виконана у вигляді *middleware* та виконується перед відправкою запиту на сервер. За рахунок локального зберігання цих даних при повторному запиті можна взяти дані з локального сховища. Система була протестована мануально та автоматизовано. Мануально був перевірений функціонал та клієнтська частина інформаційної системи. За допомогою unit-тестів було додатково протестований функціонал системи.

ПЕРЕЛІК ПОСИЛАНЬ

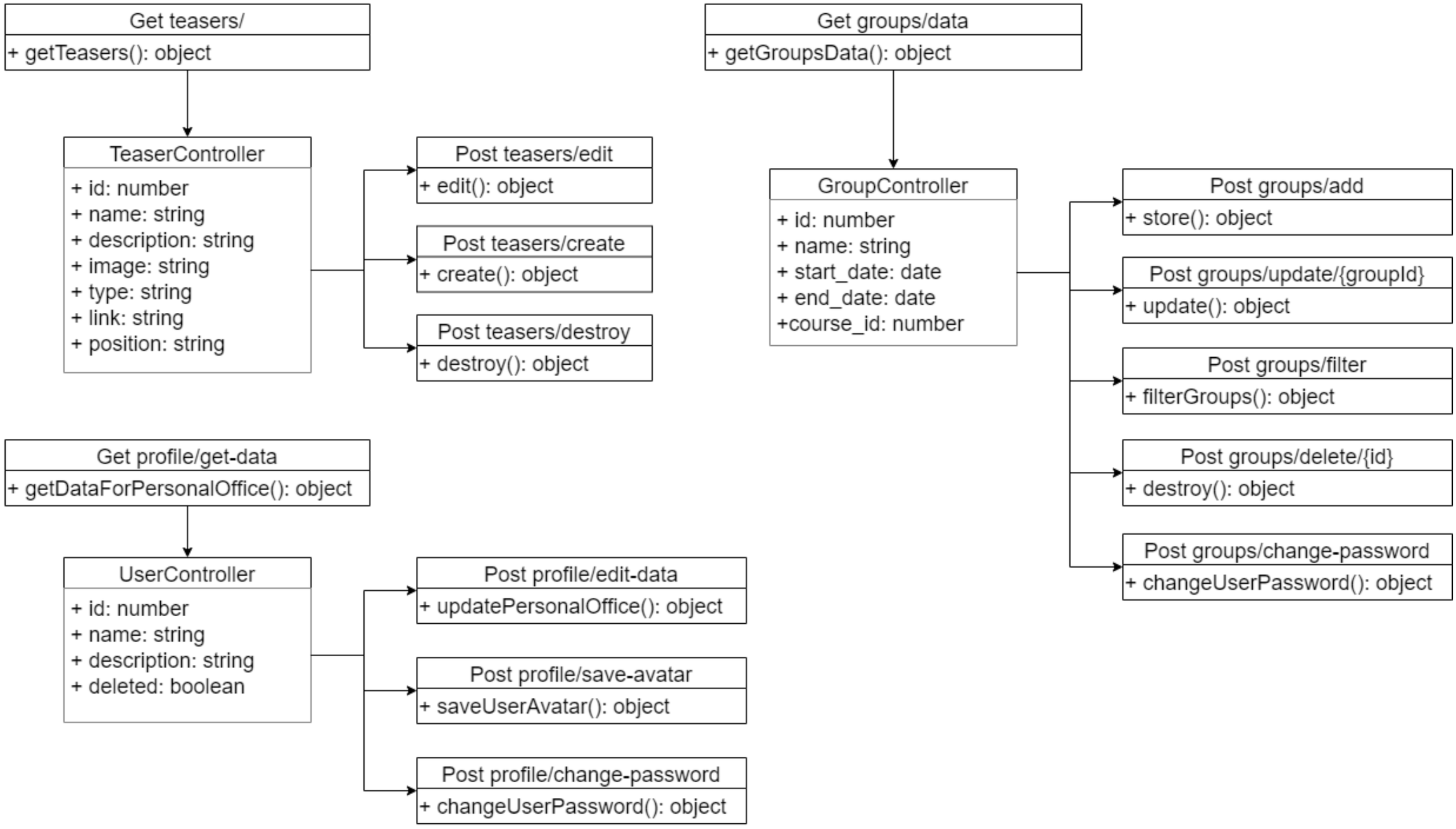
1. Дослідження UX [Електронний ресурс] : https://www.marketingdive.com/news/google-53-of-mobile-users-abandon-sites-that-take-over-3-seconds-to-load/426070/
2. Побудова інтерфейсу за допомогою F-моделі [Електронний ресурс] : https://webdesign.tutsplus.com/articles/understanding-the-f-layout-in-web-design--webdesign-687
3. Документація ExpressJS [Електронний ресурс] : https://expressjs.com/ru/
4. Документація бібліотеки React [Електронний ресурс] : https://ru.reactjs.org/
5. Документація CSS фреймворку Bootstrap [Електронний ресурс] : https://getbootstrap.com/
6. Документація CSS [Електронний ресурс] : https://ru.wikipedia.org/wiki/CSS
7. Документація SCSS [Електронний ресурс] : https://sass-scss.ru/guide/
8. Документація життєвих циклів бібліотеки React [Електронний ресурс] : https://programmingwithmosh.com/javascript/react-lifecycle-methods/
9. Документація мови програмування JavaScript [Електронний ресурс] : https://learn.javascript.ru/

ДОДАТКИ

ДОДАТОК А
Конфігураційний файл для Webpack

```
module: {
  rules: [
    {
      enforce: 'pre',
      test: /\.(js|jsx)$/,
      exclude: /node_modules/,
      loader: 'eslint-loader',
      resolve: {
        extensions: ['.js', '.jsx', '.es6'],
      },
    },
    {
      test: /\.(js|jsx)$/,
      exclude: /node_modules/,
      use: {
        loader: 'babel-loader',
      },
      resolve: {
        extensions: ['.js', '.jsx', '.es6'],
      },
    },
    {
      test: /\.scss$/,
      use: [MiniCssExtractPlugin.loader, 'css-loader', 'postcss-loader', 'sass-loader'],
    },
    {
      test: /\.svg$/,
      use: [
        {
          loader: 'svg-sprite-loader',
          options: {
            extract: true,
          },
        },
      ],
    },
    {
      test: /\.(woff(2)?|ttf|eot)(\?v=\d+\.\d+\.\d+)?$/,
      use: [{
        loader: 'file-loader',
        options: {
          name: 'fonts/[name].[ext]',
          publicPath: '../',
        },
      }],
    },
```

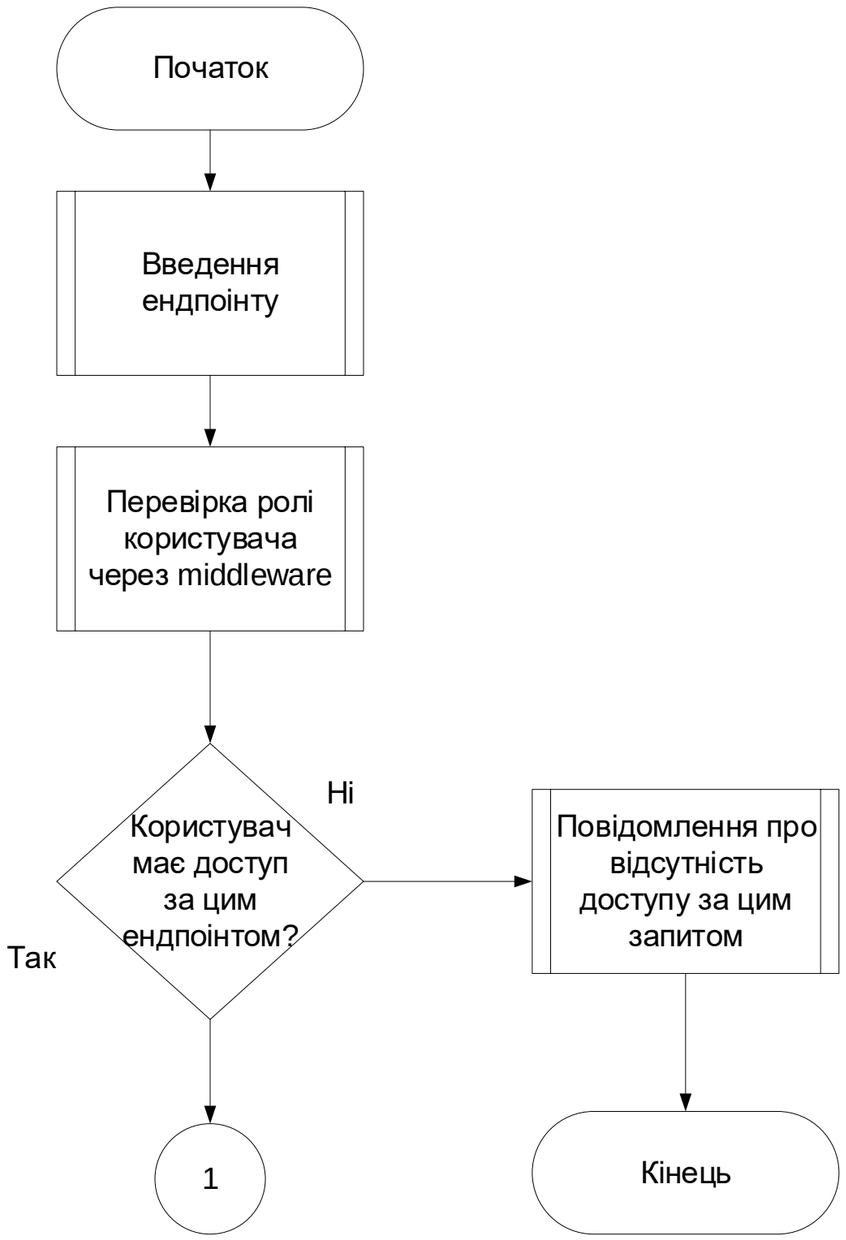
REST API

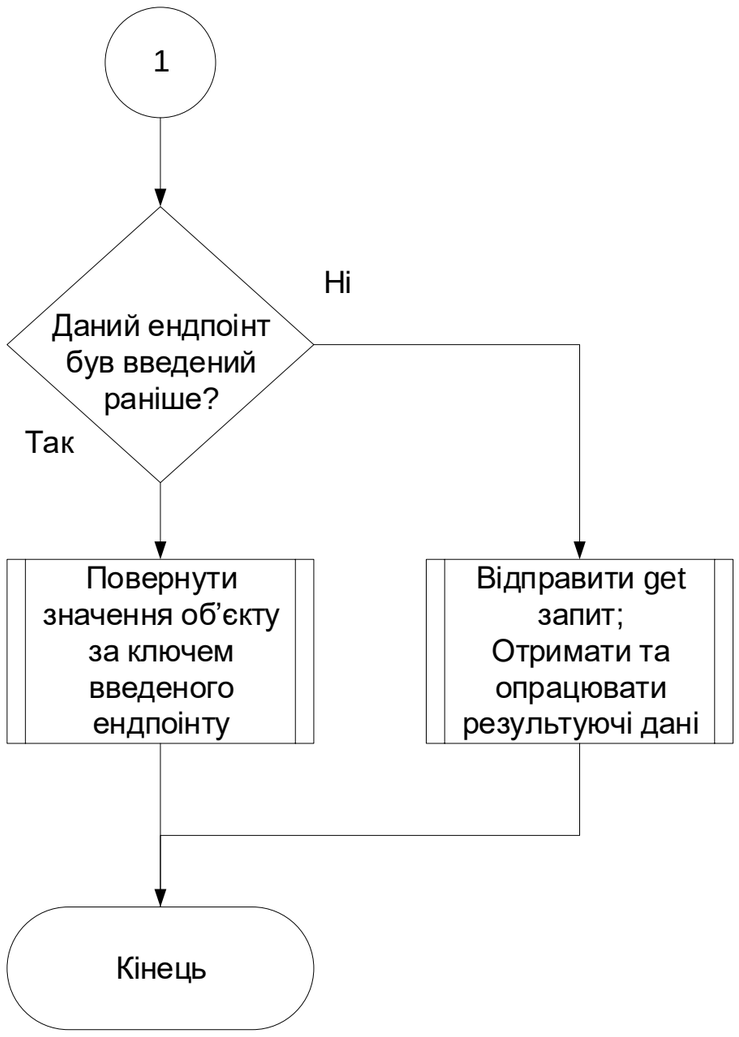


Демонстраційний плакат №1
до магістерської дисертації роботи на тему
«Рейтингова система навчання студентів з дисципліни робототехніка»

Виконав: студент гр. ІК-81мп Вєтошкін І.В.
Керівник: ст. викладач Анікін В.К.

Алгоритм кешування системи





Демонстраційний плакат №2
до магістерської дисертації на тему
«Рейтингова система навчання студентів з дисципліни робототехніка»

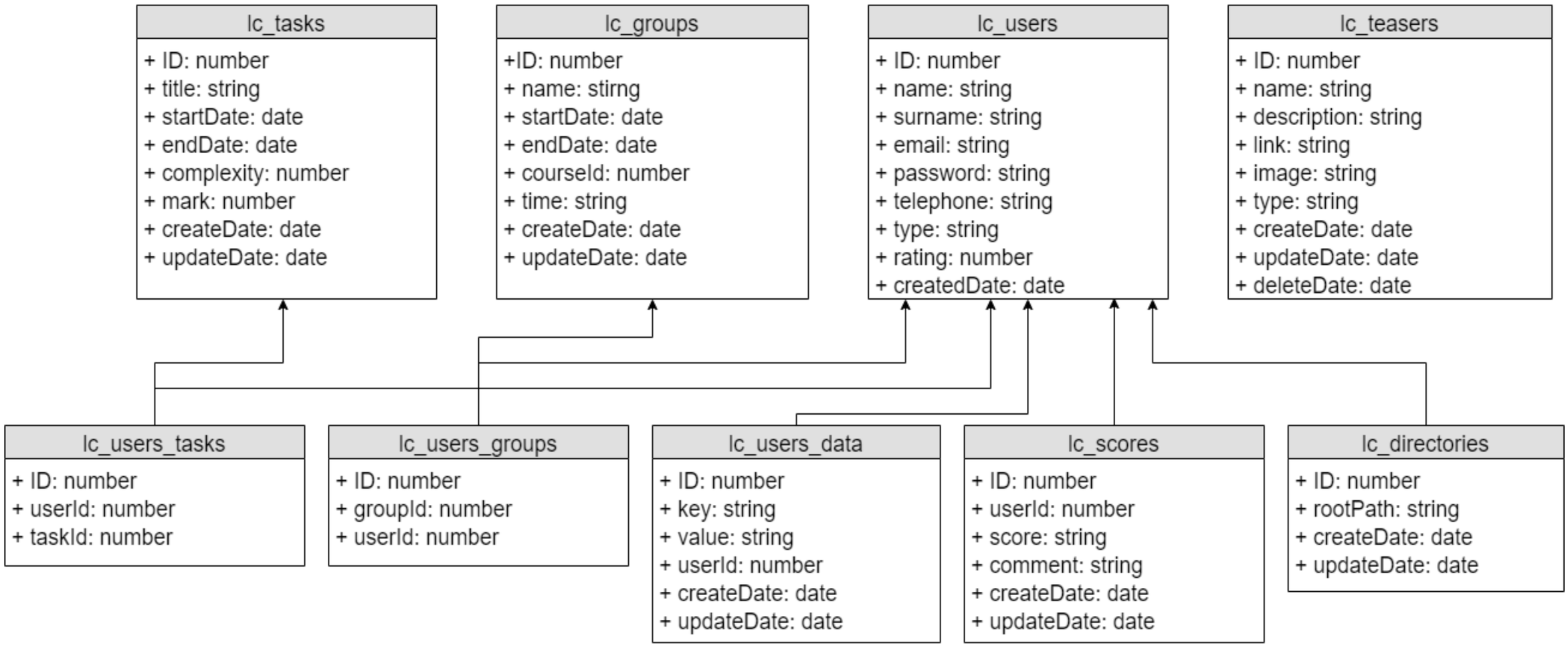
Виконав: студент гр. ІК-81мп Вєтошкін І.В.
Керівник: ст. викладач Анікін В.К.

Архітектура рейтингової системи

Демонстраційний плакат №3
до магістерської дисертації на тему
«Рейтингова система навчання студентів з дисципліни робототехніка»

Виконав: студент гр. ІК-81мп Вєтошкін І.В.
Керівник: ст. викладач Анікін В.К.

База даних рейтингової системи



Демонстраційний плакат №4
до магістерської дисертації на тему
«Рейтингова система навчання студентів з дисципліни робототехніка»

Виконав: студент гр. ІК-81мп Вєтошкін І.В.
Керівник: ст. викладач Анікін В.К.

Інтерфейс рейтингової системи

Редактирование урока

Название урока: Урок 2

Дата: 14.02.2019

Шкала оценивания

СТУДЕНТЫ ГРУППЫ

№	Фамилия	Имя	Оценка
1	Бакай	Татьяна	0
2	Косовцова	Мария	4
3	Кушнир	Артем	0
4	Пионткевич	Роман	4
5	Рак	Артем	4
6	Семенченко	Даниил	0

СОХРАНИТЬ

УСПЕВАЕМОСТЬ

Успеваемость | Подарки

3D-моделирование, ЧТ, 17:00 (Николаев Артём)

ДОБАВИТЬ УРОК | ПРОСМОТРЕТЬ ОЦЕНКИ

Урок 1	2019-02-07
Урок 2	2019-02-14
Урок 3	2019-02-21
Урок 4	2019-02-28
Урок 5	2019-03-07
Урок 6	2019-03-14

Оценки группы - 3D-моделирование, ЧТ, 17:00 (Николаев Артём)

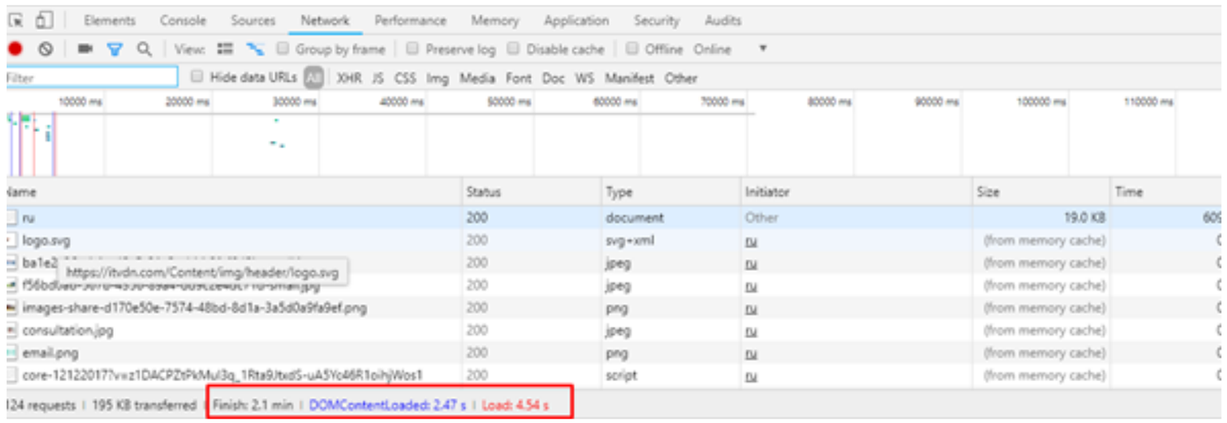
Шкала оценивания

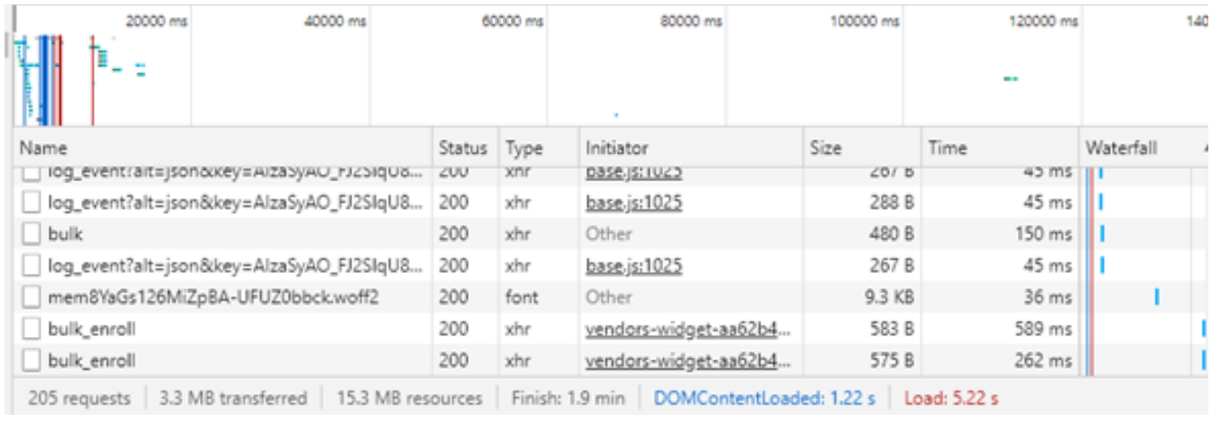
№	Фамилия	Имя	Сумма	2019-02-07	2019-02-14	2019-02-21	2019-02-28	2019-03-07	2019-03-14	2019-03-21	2019-03-28	2019-04-04	2019-04-11	2019-0
1	Бакай	Татьяна	22	0	0	0	0	4	0	5	0	0	0	0
2	Косовцова	Мария	81	3	4	5	5	4	4	5	5	5	3	3
3	Кушнир	Артем	19	0	0	0	0	0	0	5	0	0	0	0
4	Пионткевич	Роман	64	3	4	5	3	4	4	5	0	5	3	3
5	Рак	Артем	77	3	4	5	5	4	4	5	4	5	3	3
6	Семенченко	Даниил	25	0	0	0	0	4	0	5	0	0	0	0

Демонстраційний плакат №5
до магістерської дисертації на тему
«Рейтингова система навчання студентів з дисципліни робототехніка»

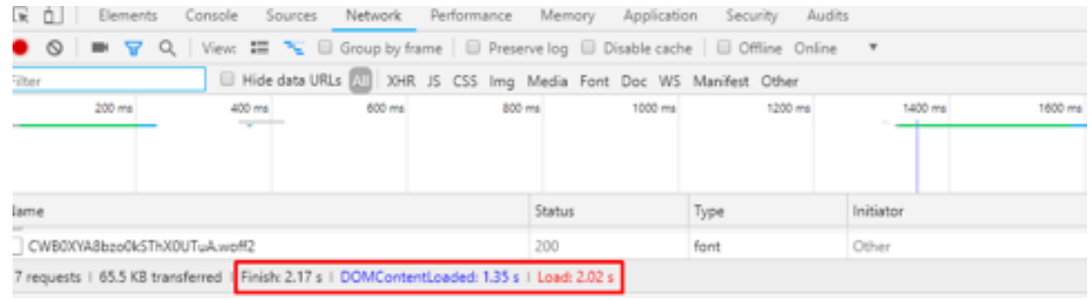
Виконав: студент гр. ІК-81мп Вєтошкін І.В.
Керівник: ст. викладач Анікін В.К.

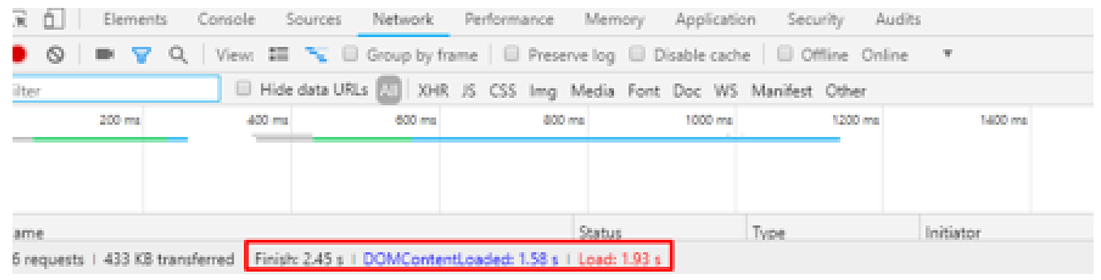
Показники швидкодії існуючих систем аналогів





Показники швидкодії розробленої системи





Демонстраційний плакат №6
до магістерської дисертації на тему
«Рейтингова система навчання студентів з дисципліни робототехніка»

Виконав: студент гр. ІК-81мп Вєтошкін І.В.
Керівник: ст. викладач Анікін В.К.